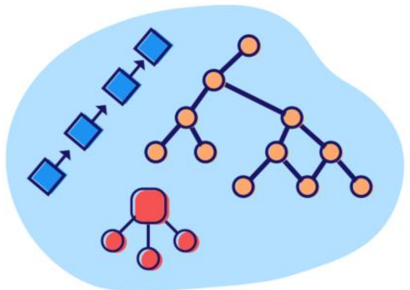
الجامعة العربية الخاصة للعلوم والتكنولوجيا
كلية الهندسة المعلوماتية
قسم تقانة المعلومات

Algorithms & Data Structures (2)

Lecture-3: Graph
(البيان)



Dr. Asmaa Shaar

فهرس المحاضرة

- شجرة الامتداد الاصغرية (Minimum Spanning Tree):
  - خوارزمية Kruskal
  - خوارزمية Prim
- Topological Sort
- مسألة المسار الأقصر
  - خوارزمية Dijkstra

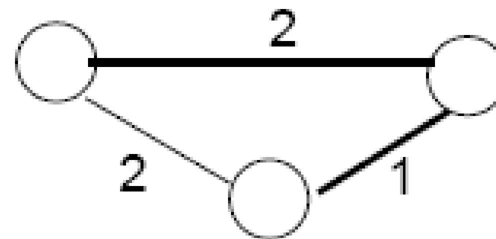
أشجار الامتداد الاصغرية

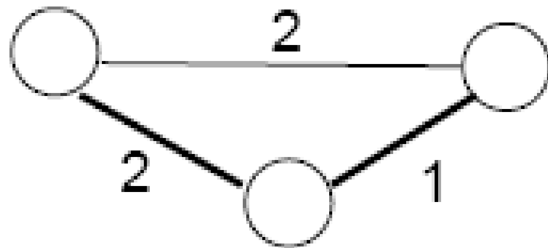
- شجرة الامتداد (Spanning Tree): عبارة عن شجرة (بيان متصل لاحلقي) مكونة من الاضلاع الشجرية وجميع رؤوس البيان.

- غابة الامتداد (Spanning forest): اذا كان البيان **غير متصل**، عندها هناك شجرة امتداد لكل مكونة متصلة (connected component) في البيان.

- نعرف كلفة شجرة الامتداد لبيان **موزون** غير موجه **بمجموع اوزان الاضلاع** في شجرة الامتداد.

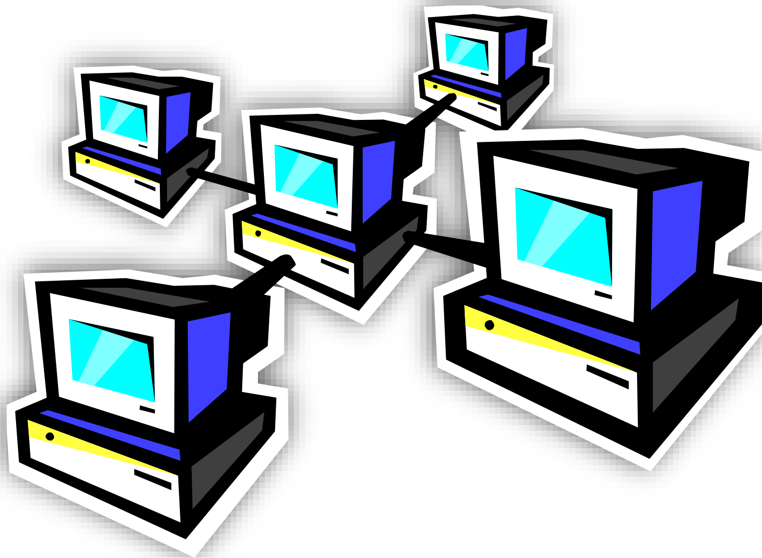
- نعرف شجرة الامتداد الاصغرية (minimum spanning tree) أو اختصاراً MST بأنها شجرة الامتداد **ذات الكلفة الاقل** وهي ليست بالضرورة **وحيدة**.

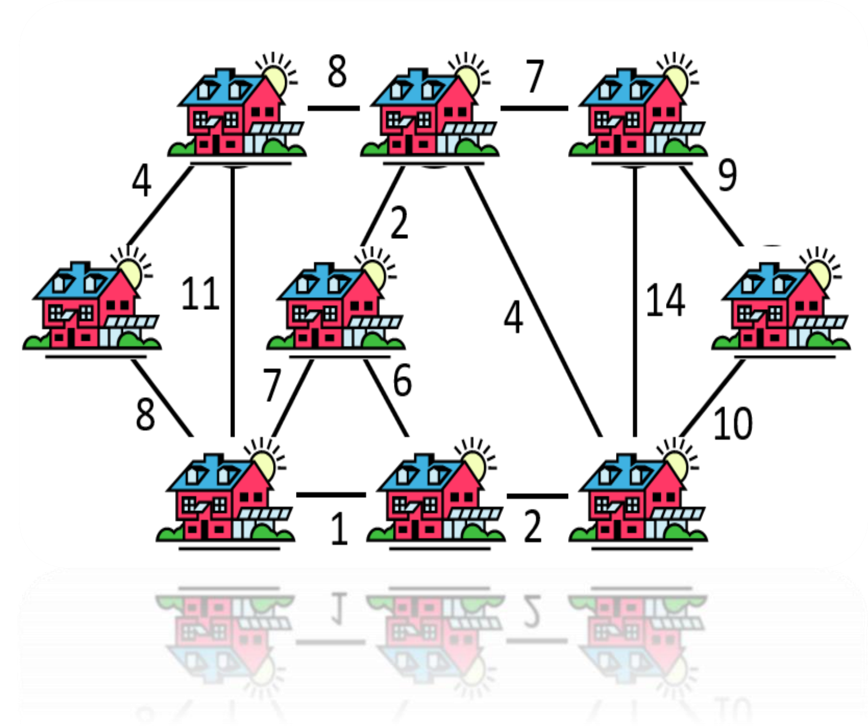




من أمثلة تطبيقات MST

- **مثال 1:** ايجاد الطريقة **الاقل كلفة** لربط مجموعة من الحواسيب الموجودة في مكاتب شركة ما.
- **مثال 2:** ايجاد **كلفة اصلاح اجمالية اصغرية** لمجموعة طرق تصل بين مجموعة منازل (مع امكانية الوصول لكل منزل من المنازل الاخرى).





أشجار الامتداد الاصغرية

- بشكل عام، تكون مسألة MST:
  - يكون لدينا بيان موزون غير موجه: لكل ضلع $E \ni (u, v)$ وزن $w(u, v)$
  - **المطلوب:** ايجاد $T \subseteq E$ بحيث **T** مكونة من **$n-1$** ضلع تربط **جميع رؤوس** البيان **بأقل كلفة ممكنة** ($w(T) = \Sigma_{(u,v)\in T}\, w(u, v)$ is minimized).

- من الخوارزميات المستخدمة لايجاد MST:
  - خوارزمية Kruskal
  - خوارزمية Prim

- تعتمد هذه الخوارزميات على **نهج غريدي** الذي فيه يتم بناء الحل الامثلي على **عدة مراحل** و في كل مرحلة يتم اختيار **أفضل قرار** حسب معيار ما معين (في حالتنا least cost).

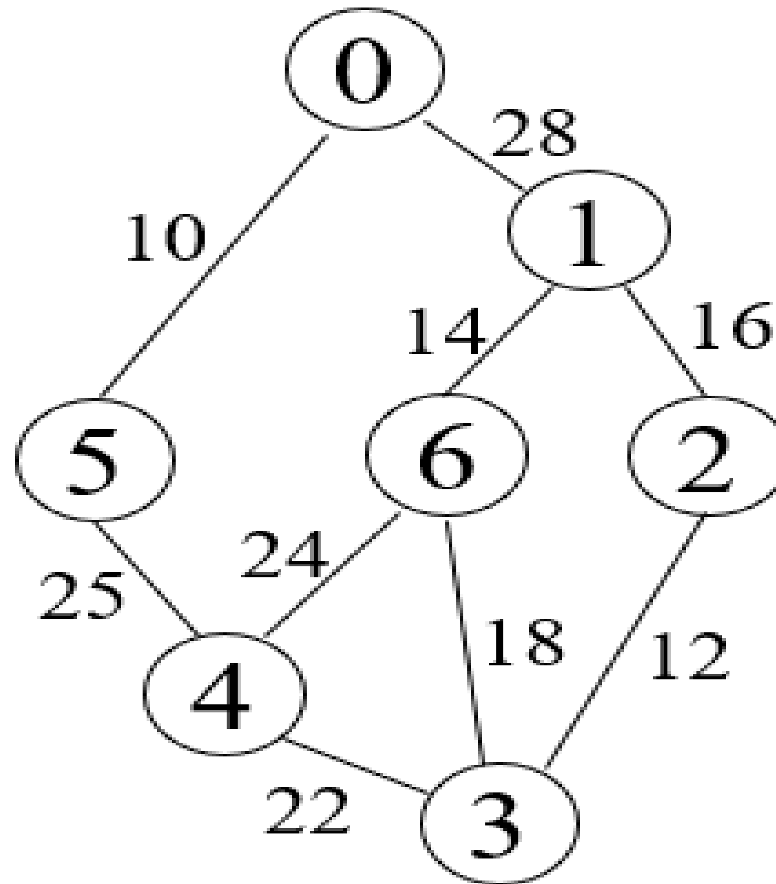
Kruskal خوارزمية

```
T= { };
while (T contains less than n-1 edges
        && E is not empty)
{   choose a least weight edge (v,w) from E;
    delete (v,w) from E;
    if ((v,w) does not create a cycle in T)
          add (v,w) to T
    else
          discard (v,w);
}
if (T contains fewer than n-1 edges)
  printf("No MST\n");
```

- في هذه الخوارزمية:
  - يتم بناء شجرة الامتداد الاصغرية T من خلال **اضافة ضلع واحد فقط** لـ T في **كل خطوة**.
  - يتم **اختيار** ضلع لتضمنيه ضمن T بترتيب تصاعدي حسب الوزن بحيث لا يؤدي إلى تشكيل حلقة.

خوارزمية Kruskal

- **مثال1:** بفرض لدينا البيان الموزون غير الموجه التالي والمطلوب ايجاد MST حسب خوارزمية Kruskal



خوارزمية Kruskal

0 —10— 5

2 —12— 3

1 —14— 6

1 —16— 2

3 —18— 6

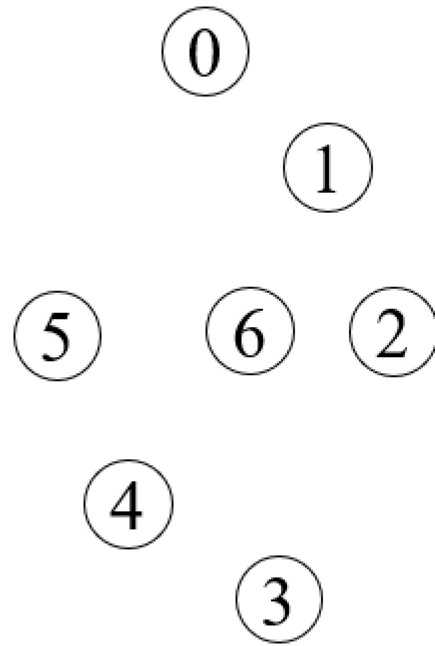
3 —22— 4

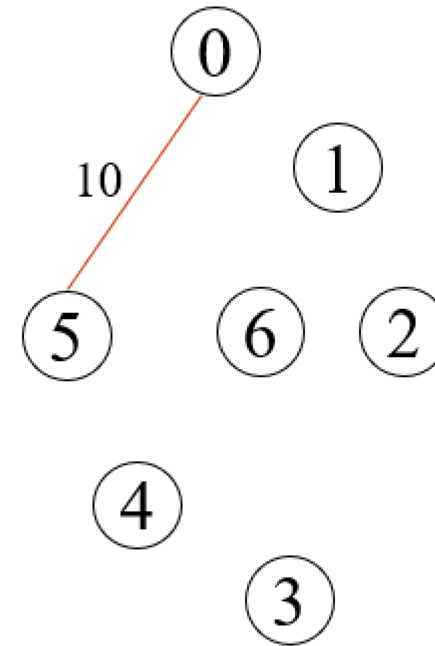
4 —24— 6

4 —25— 5

0 —28— 1

- **مثال1:** بفرض لدينا البيان الموزون غير الموجه التالي والمطلوب ايجاد MST حسب خوارزمية Kruskal

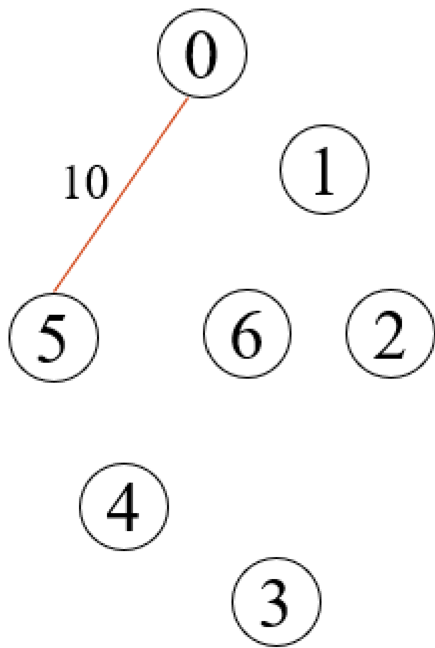


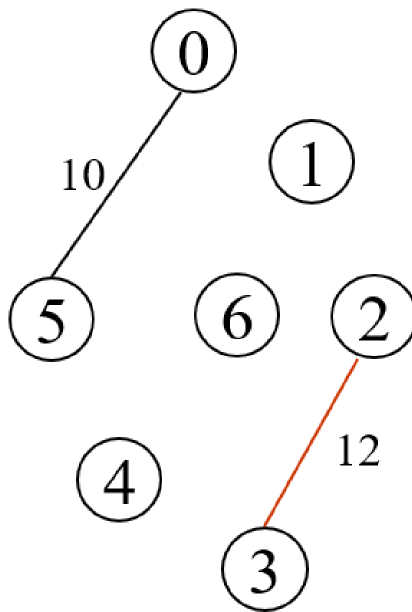


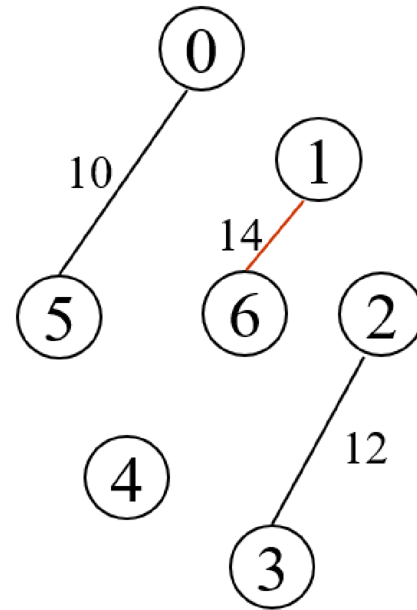
خوارزمية Kruskal

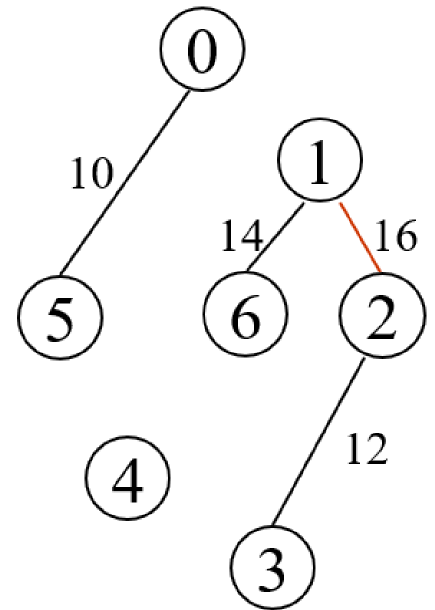
0	10	5
2	12	3
1	14	6
1	16	2
3	18	6
3	22	4
4	24	6
4	25	5
0	28	1

- **مثال1:** بفرض لدينا البيان الموزون غير الموجه التالي والمطلوب ايجاد MST حسب خوارزمية Kruskal





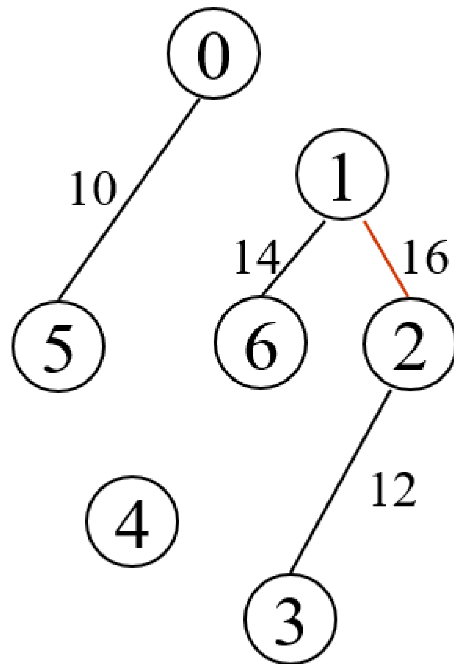




خوارزمية Kruskal

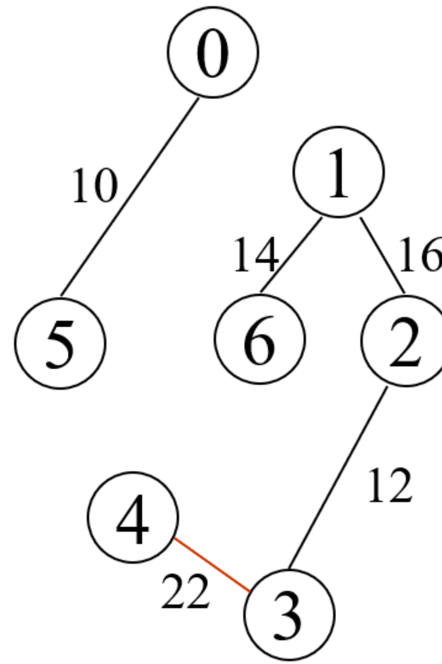
- **مثال1:** بفرض لدينا البيان الموزون غير الموجه التالي والمطلوب ايجاد MST حسب خوارزمية Kruskal

0 —10— 5
2 —12— 3
1 —14— 6
1 —16— 2
3 —18— 6
3 —22— 4
4 —24— 6
4 —25— 5
0 —28— 1



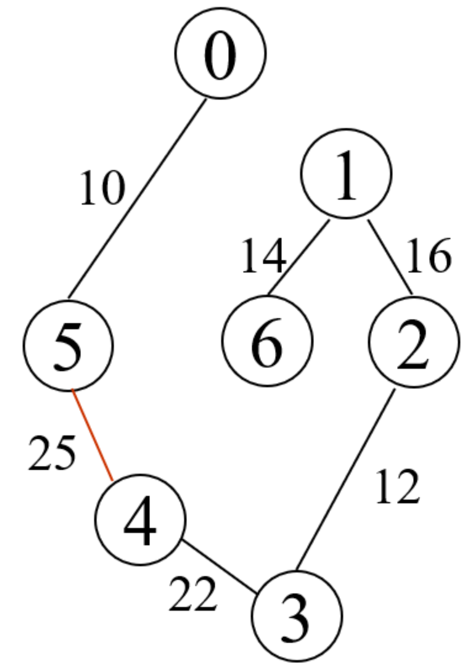
\+ 3 — 6

cycle



\+ 4 — 6

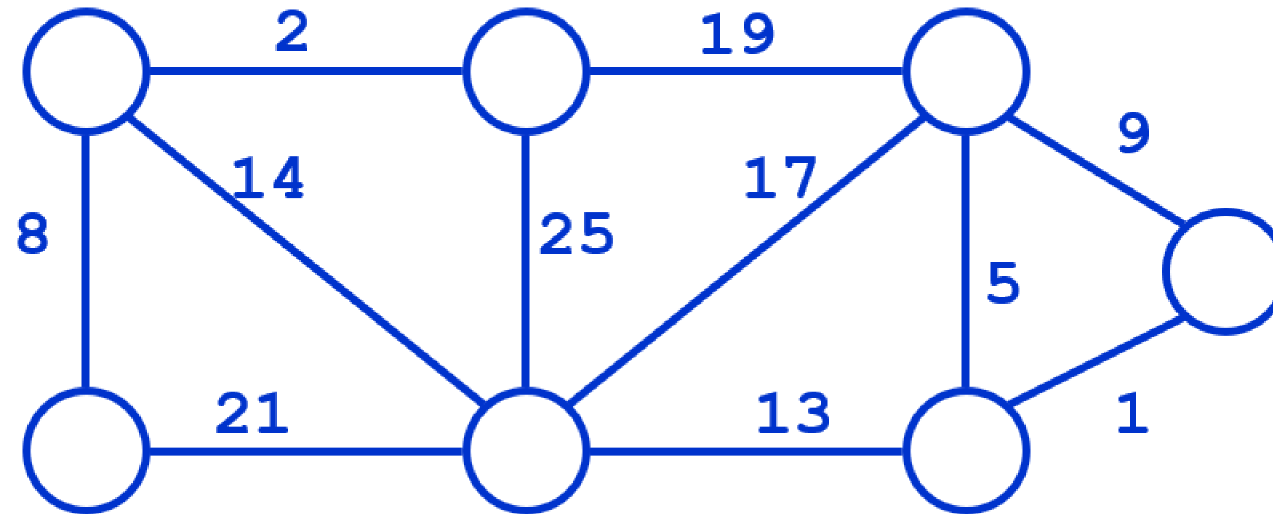
cycle



cost = 10 +25+22+12+16+14

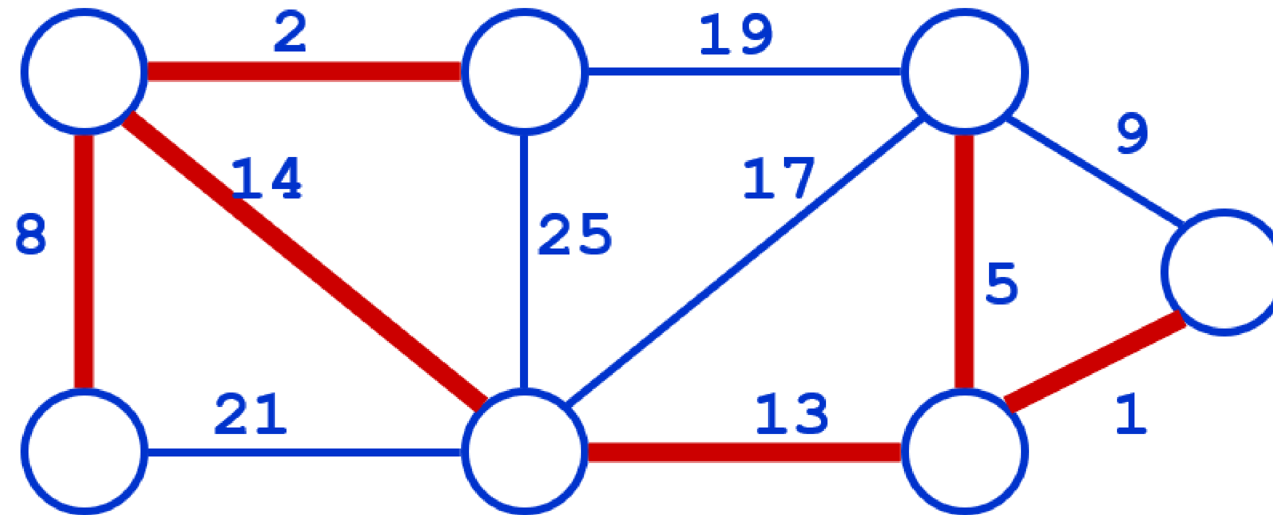
خوارزمية Kruskal

- **مثال2:** بفرض لدينا البيان الموزون غير الموجه التالي والمطلوب ايجاد MST حسب خوارزمية Kruskal



خوارزمية Kruskal

- **مثال2:** بفرض لدينا البيان الموزون غير الموجه التالي والمطلوب ايجاد MST حسب خوارزمية Kruskal



خوارزمية Prim

```
T={ };
TV={0};
while (T contains fewer than n-1 edges)
    {
      let (u,v) be a least cost edge such
           that u ∈ TV and v ∉ TV
      if (there is no such edge ) break;
      add (u,v) to T;
      add v to TV;
    }
if (T contains fewer than n-1 edges)
  printf("No MST\n");
```

تتضمن المجموعة **TV** **العقد** التي تم تضمنيها فيMST ، ويتم البدء من **عقدة اختيارية** ما

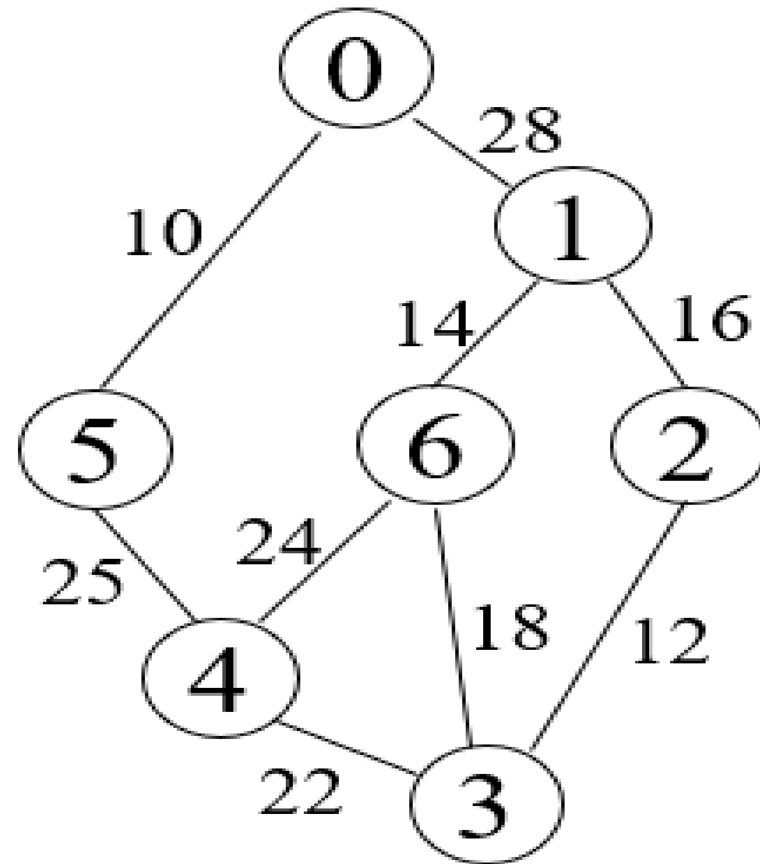
في كل خطوة يتم الاخذ بعين الاعتبار الاضلاع التي **تربط بين عقد** المجموعة **TV** وبقية عقد البيان ويتم اختيار الضلع (u,v) الذي يمتلك **الوزن الاقل** لتضمنيه ضمن T ونقل العقدة v للمجموعة **TV**

في هذه الخوارزمية:

- يتم بناء شجرة الامتداد الاصغرية T من خلال اضافة **ضلع واحد فقط** لـ T في **كل خطوة**.
- دائماً تشكل الاضلاع التي في المجموعة T **شجرة وحيدة**.

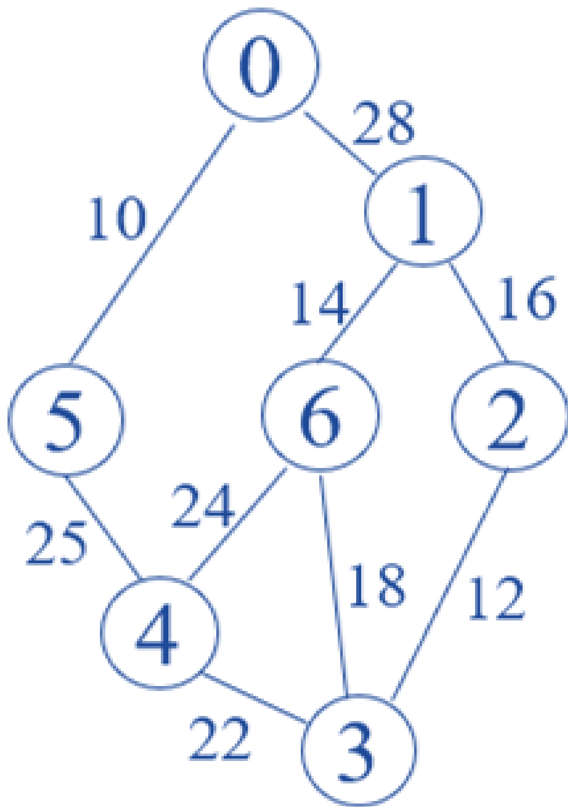
خوارزمية Prim

- **مثال1:** بفرض لدينا البيان الموزون غير الموجه التالي والمطلوب ايجاد MST حسب خوارزمية Prim.

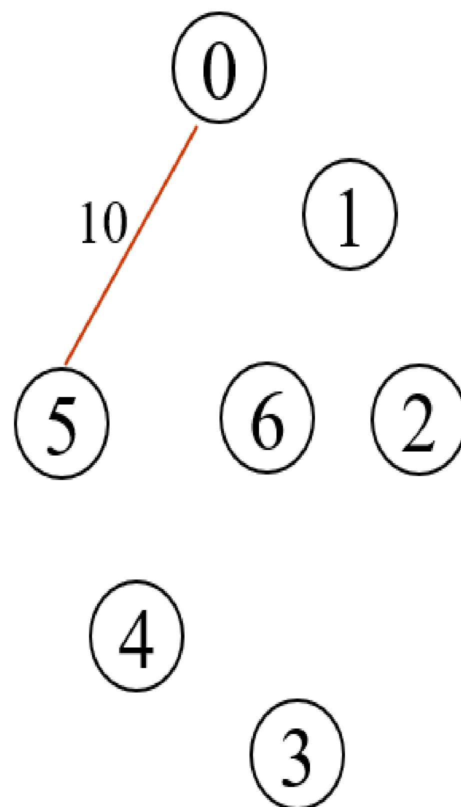


خوارزمية Prim

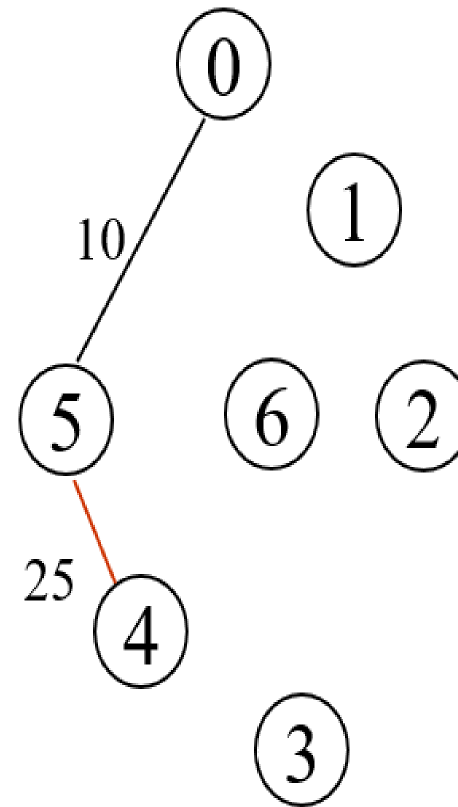
- **مثال1:** بفرض لدينا البيان الموزون غير الموجه التالي والمطلوب ايجاد MST حسب خوارزمية Prim.



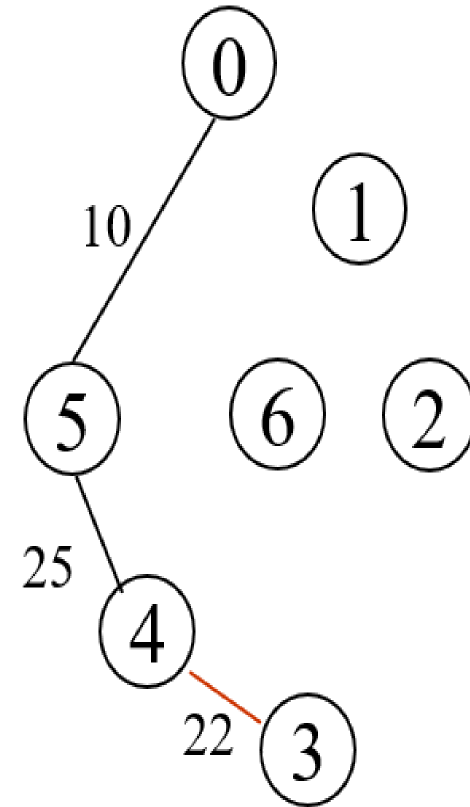
TV={0};



TV={0,5};



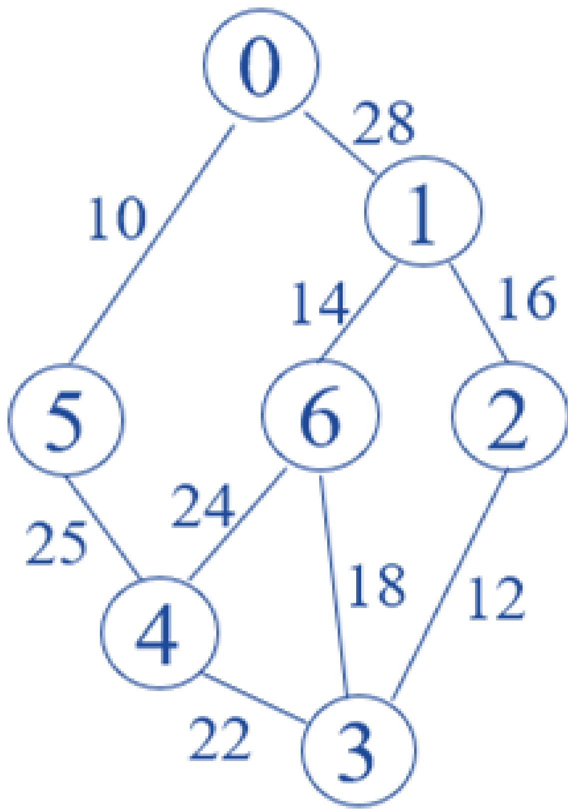
TV={0,5,4};



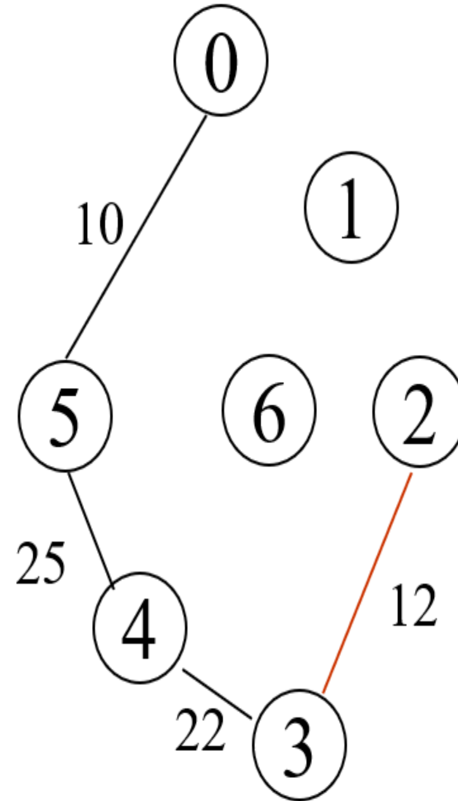
TV={0,5,4,3};

خوارزمية Prim

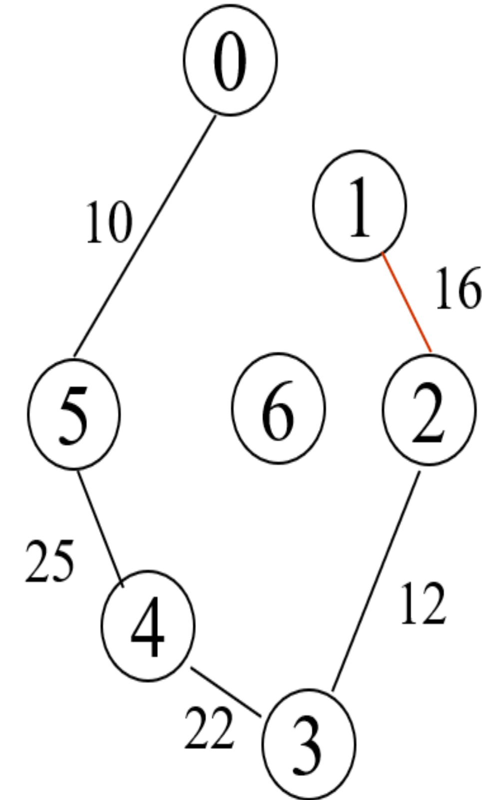
- **مثال1:** بفرض لدينا البيان الموزون غير الموجه التالي والمطلوب ايجاد MST حسب خوارزمية Prim.



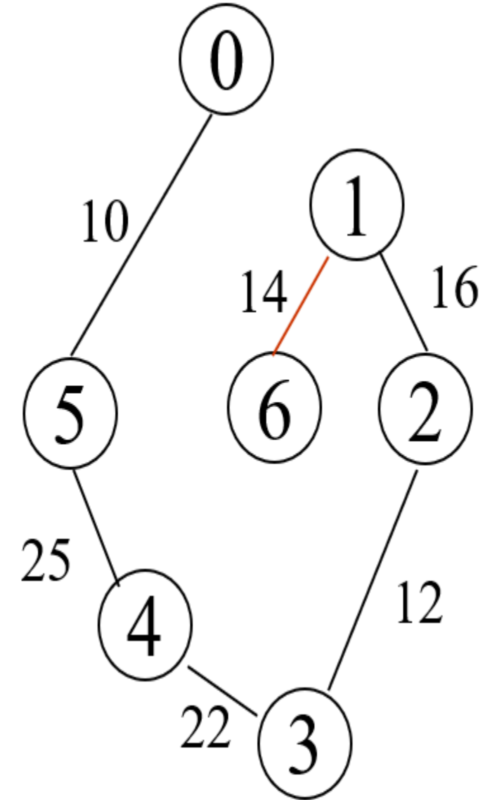
TV={0,5,4,3};



TV={0,5,4,3,2};



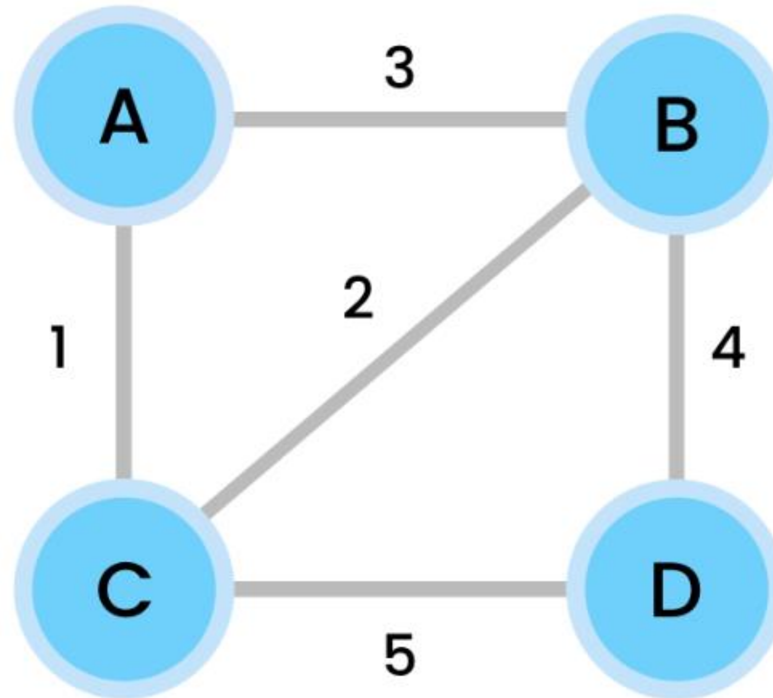
TV={0,5,4,3,2,1};



TV={0,5,4,3,2,1,6};

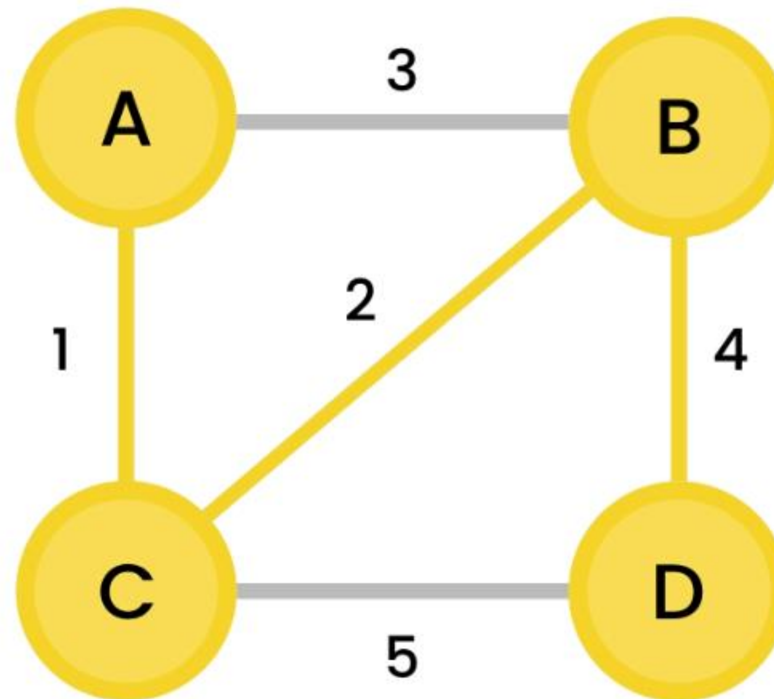
خوارزمية Prim

- **مثال2:** بفرض لدينا البيان الموزون غير الموجه التالي والمطلوب ايجاد MST حسب خوارزمية Prim.



خوارزمية Prim

- **مثال2:** بفرض لدينا البيان الموزون غير الموجه التالي والمطلوب ايجاد MST حسب خوارزمية Prim.



Prim vs. Kruskal

- في خوارزمية Prim: دائماً هناك **شجرة وحيدة** تتمو
  - أي تتشكل الاضلاع التي في المجموعة T **شجرة وحيدة** دائماَ.

- في خوارزمية Kruskal: هناك **عدة اشجار** تنمو في نفس الوقت بحيث يتم دمج هذه الأشجار باستخدام الاضلاع المضافة.
  - أي بعد n - 1 خطوة سوف يكون لدينا **شجرة وحيدة**.

الفرز الطوبولوجي
(Topological Sort)

الفرز الطوبولوجي

- بشكل عام، يتم استخدام الفرز الطوبولوجي في بيان **موجه لاحلقي (DAG)** يحتوي على **علاقات اعتمادية** بين العقد (dependency graph) بهدف تحديد الترتيب المسموح للعقد.
  - يتم ترتيب العقد بشكل لا تظهر فيه عقدة قبل أي عقدة اخرى تمتلك ضلع وارد اليها (متطلب سابق)

- أمثلة عن استخدام الفرز الطوبولوجي.
  - لتحديد ترتيب الترجمة (compile) لمجموعة من الملفات المعتمدة على بعضها.
  - لتحديد الترتيب الممكن لمجموعة المقرارات التي تخص درجة جامعية ما.

- من أجل DAG ما معين، هناك **عدة ترتيبات ممكنة للعقد** باستخدام الفرز الطوبولوجي.
  - في حالة List يكون لدينا **ترتيب وحيد**.

الفرز الطوبولوجي

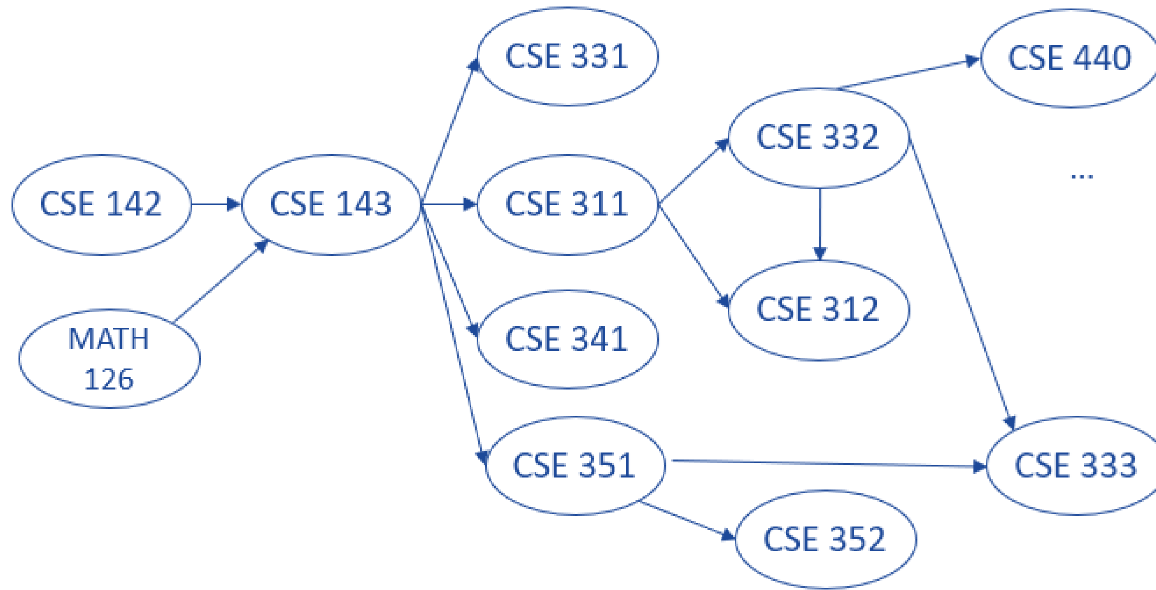
- **الطريقة العامة:**

1. نقوم بتسمية كل رأس **بالدرجة الواردة** الموافقة له (in-degree).
2. طالما لدينا هناك عقد:
   - نقوم بحذف العقدة **v** من البيان التي تمتلك درجة **واردة صفر**.
   - من أجل كل عقدة **u** مجاورة لـ **v**، نقوم بانقاص الدرجة الواردة للعقد **u.**

الفرز الطوبولوجي

- مثال (1):

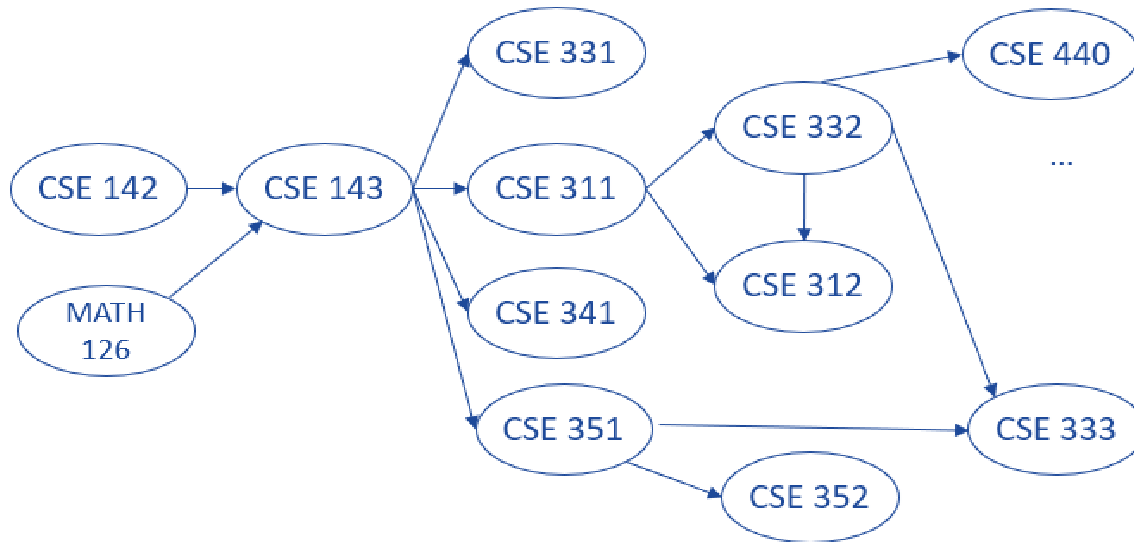
Output:



الفرز الطوبولوجي

- مثال (1):

Output:



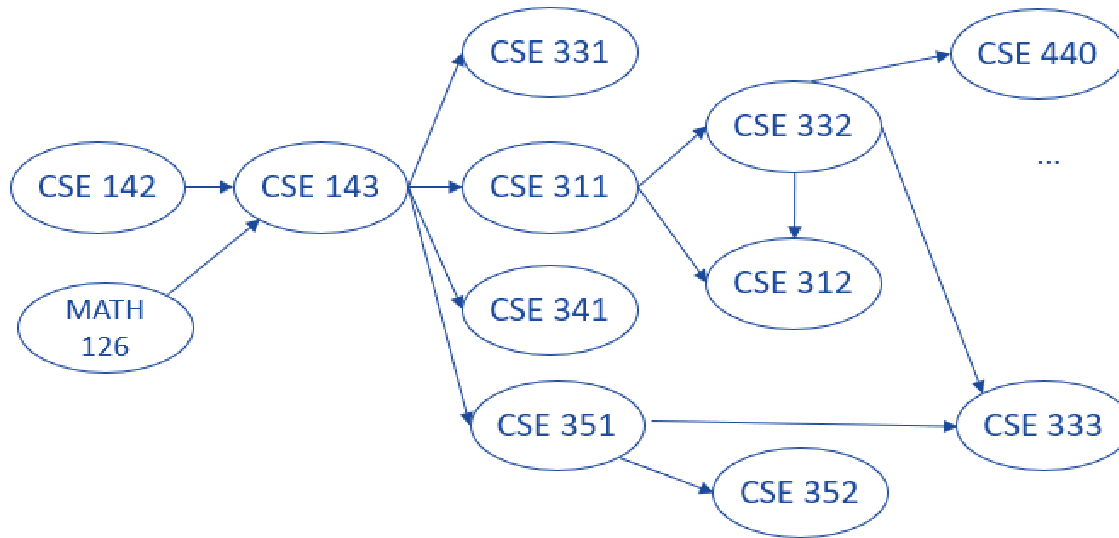
Node:	126	142	143	311	312	331	332	333	341	351	352	440
Removed?												
In-deg:	0	0	2	1	2	1	1	2	1	1	1	1

الفرز الطوبولوجي

- مثال (1):

Output:

126



Node:	126	142	143	311	312	331	332	333	341	351	352	440
Removed?	x											
In-deg:	0	0	2	1	2	1	1	2	1	1	1	1
			1									

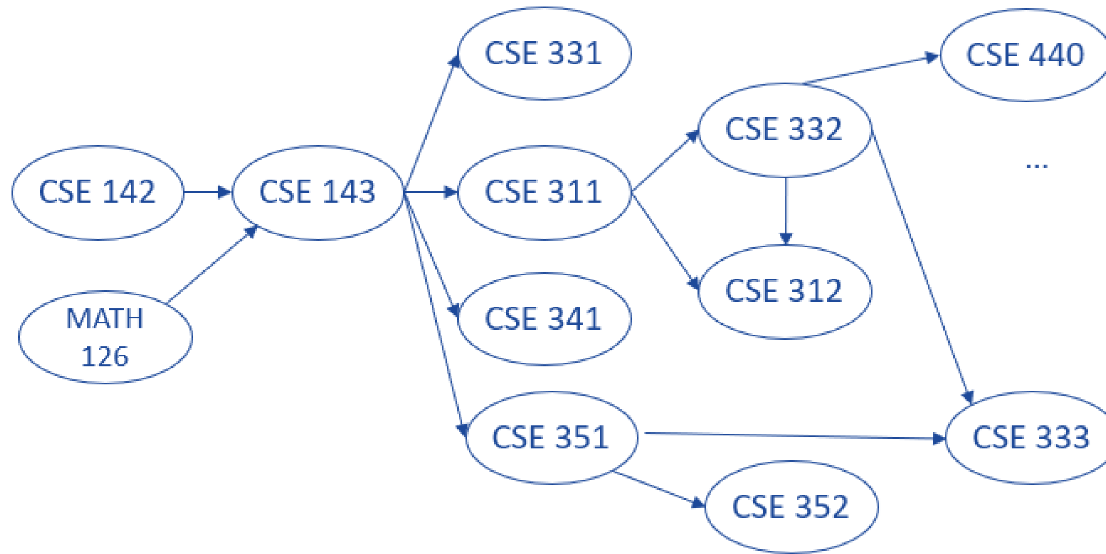
الفرز الطوبولوجي

- مثال (1):

Output:

126

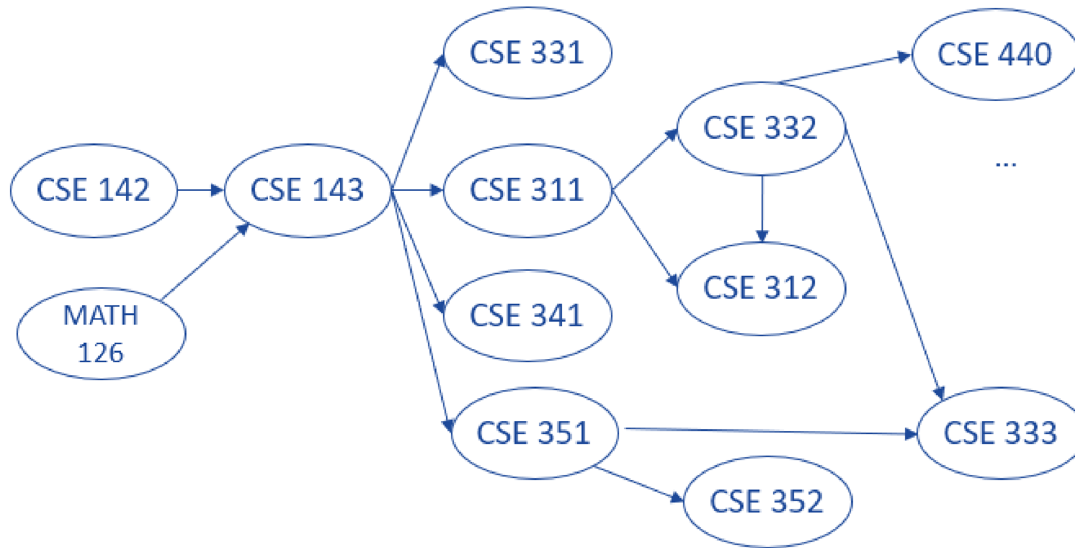
142



Node:	126	142	143	311	312	331	332	333	341	351	352	440
Removed?	x	x										
In-deg:	0	0	2	1	2	1	1	2	1	1	1	1
			1									
			0									

الفرز الطوبولوجي

- مثال (1):



Output:

126

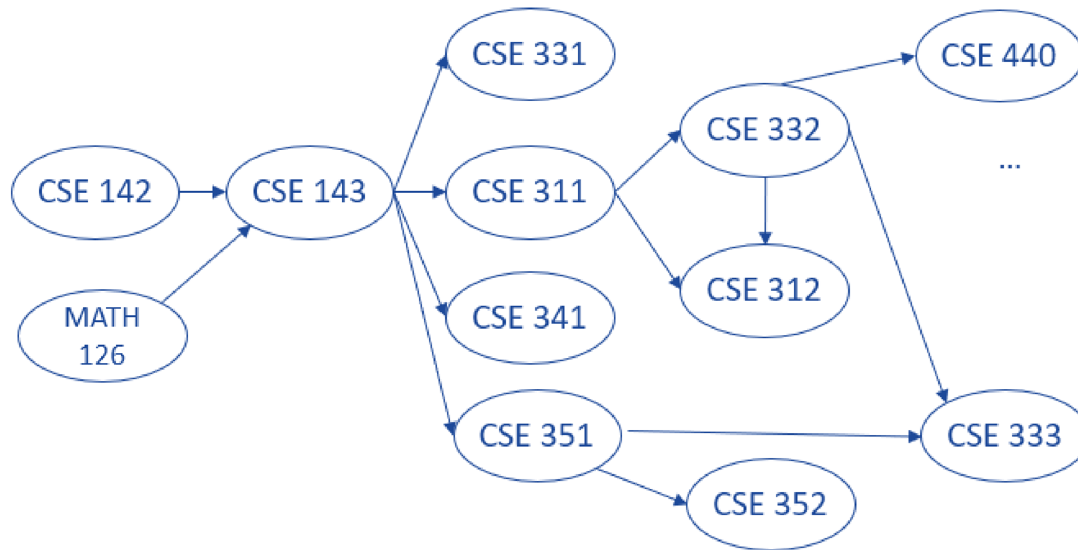
142

143

Node:	126	142	143	311	312	331	332	333	341	351	352	440
Removed?	x	x	x									
In-deg:	0	0	2	1	2	1	1	2	1	1	1	1
			1	0		0			0	0		
			0									

الفرز الطوبولوجي

- مثال (1):



Output:

126

142

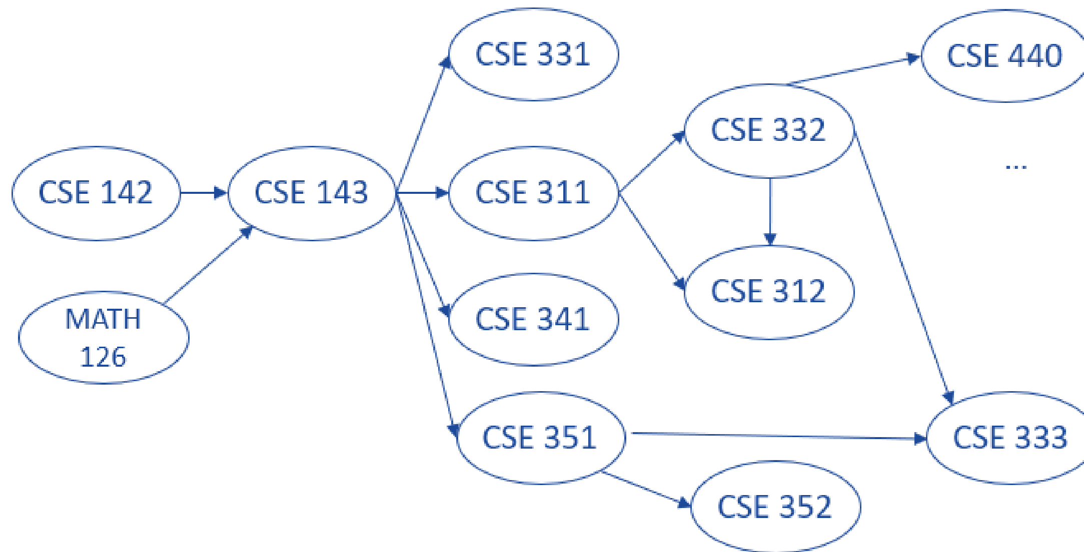
143

311

Node:	126	142	143	311	312	331	332	333	341	351	352	440
Removed?	x	x	x	x								
In-deg:	0	0	2	1	2	1	1	2	1	1	1	1
			1	0	1	0	0		0	0		
			0									

الفرز الطوبولوجي

- مثال (1):



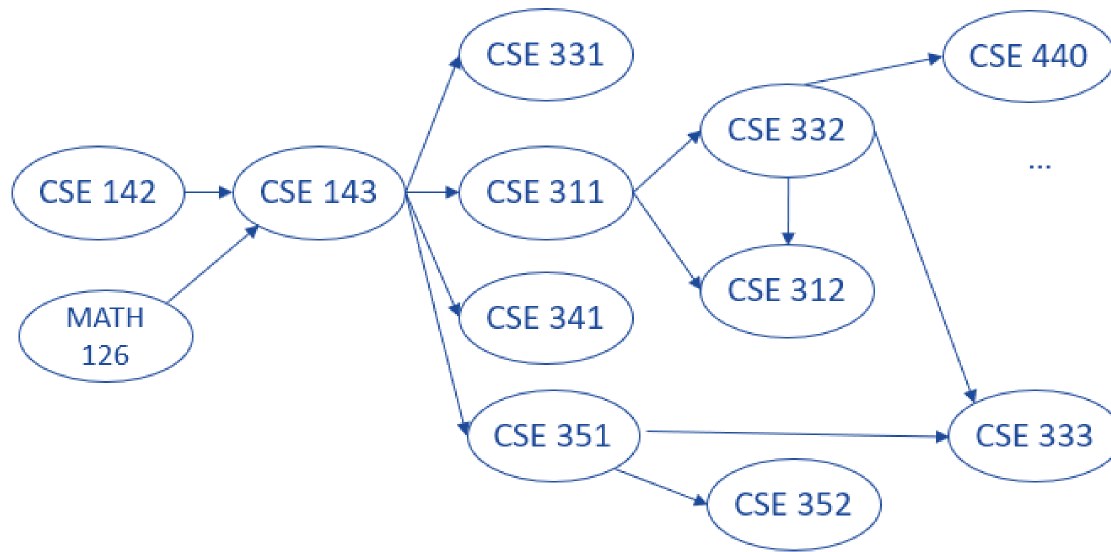
Output:

126
142
143
311
331

Node:	126	142	143	311	312	331	332	333	341	351	352	440
Removed?	x	x	x	x		x						
In-deg:	0	0	2	1	2	1	1	2	1	1	1	1
			1	0	1	0	0		0	0		
			0									

Topological Sort

- مثال (1):



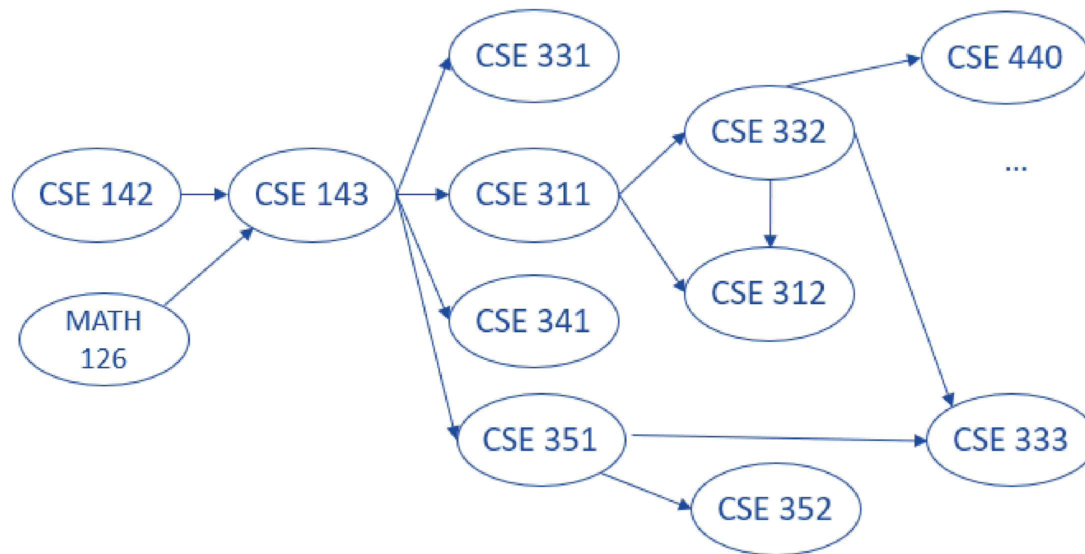
Output:

126
142
143
311
331
332

Node:	126	142	143	311	312	331	332	333	341	351	352	440
Removed?	x	x	x	x		x	x					
In-deg:	0	0	2	1	2	1	1	2	1	1	1	1
			1	0	1	0	0	1	0	0		0
			0		0							

الفرز الطوبولوجي

- مثال (1):

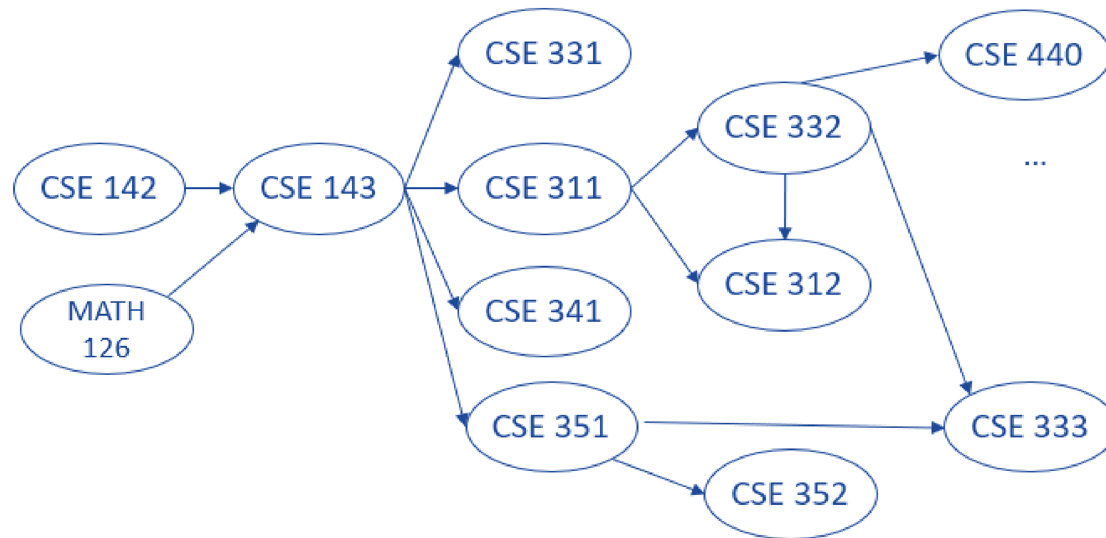


Output:
126
142
143
311
331
332
312

Node:	126	142	143	311	312	331	332	333	341	351	352	440
Removed?	x	x	x	x	x	x	x					
In-deg:	0	0	2	1	2	1	1	2	1	1	1	1
			1	0	1	0	0	1	0	0		0
			0		0							

الفرز الطوبولوجي

- مثال (1):



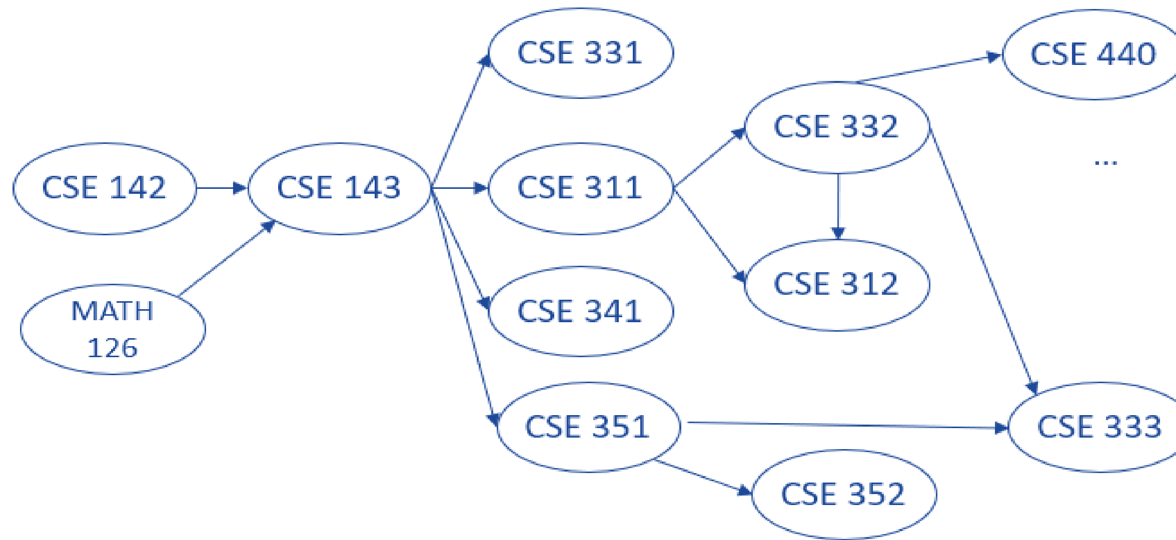
Output:

126
142
143
311
331
332
312
341

Node:	126	142	143	311	312	331	332	333	341	351	352	440
Removed?	x	x	x	x	x	x	x		x			
In-deg:	0	0	2	1	2	1	1	2	1	1	1	1
			1	0	1	0	0	1	0	0		0
			0		0							

الفرز الطوبولوجي

- مثال (1):

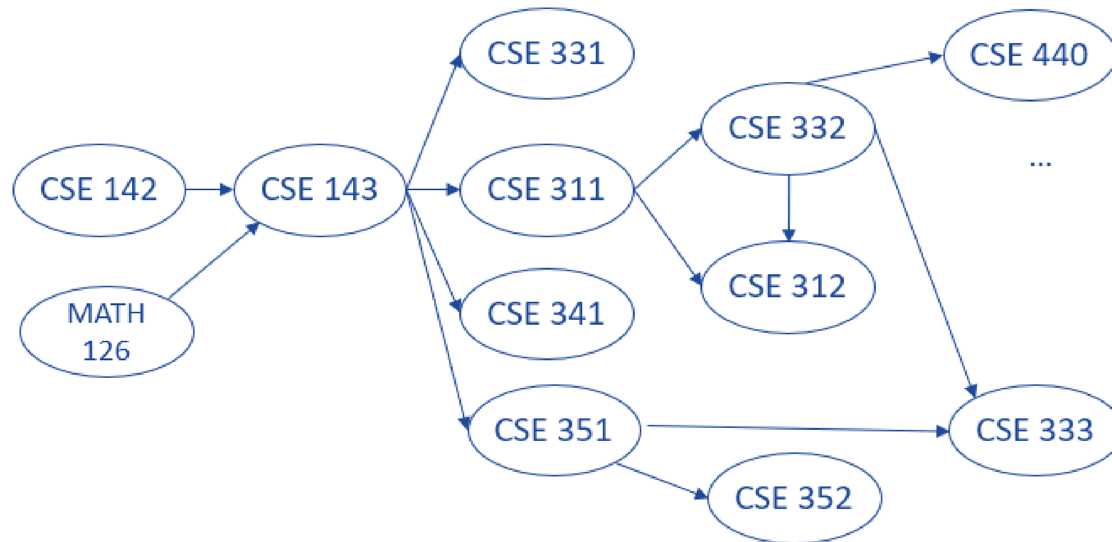


Output:
126
142
143
311
331
332
312
341
351

Node:	126	142	143	311	312	331	332	333	341	351	352	440
Removed?	x	x	x	x	x	x	x		x	x		
In-deg:	0	0	2	1	2	1	1	2	1	1	1	1
			1	0	1	0	0	1	0	0	0	0
			0		0			0				

الفرز الطوبولوجي

- مثال (1):

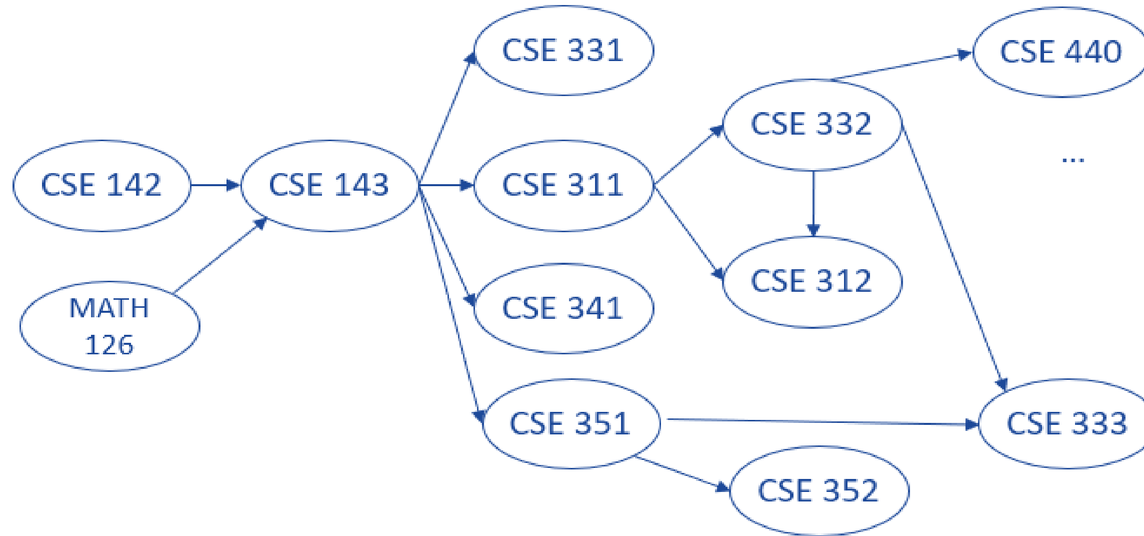


Output:
126
142
143
311
331
332
312
341
351
333

Node:	126	142	143	311	312	331	332	333	341	351	352	440
Removed?	x	x	x	x	x	x	x	x	x	x		
In-deg:	0	0	2	1	2	1	1	2	1	1	1	1
			1	0	1	0	0	1	0	0	0	0
			0		0			0				

الفرز الطوبولوجي

- مثال (1):



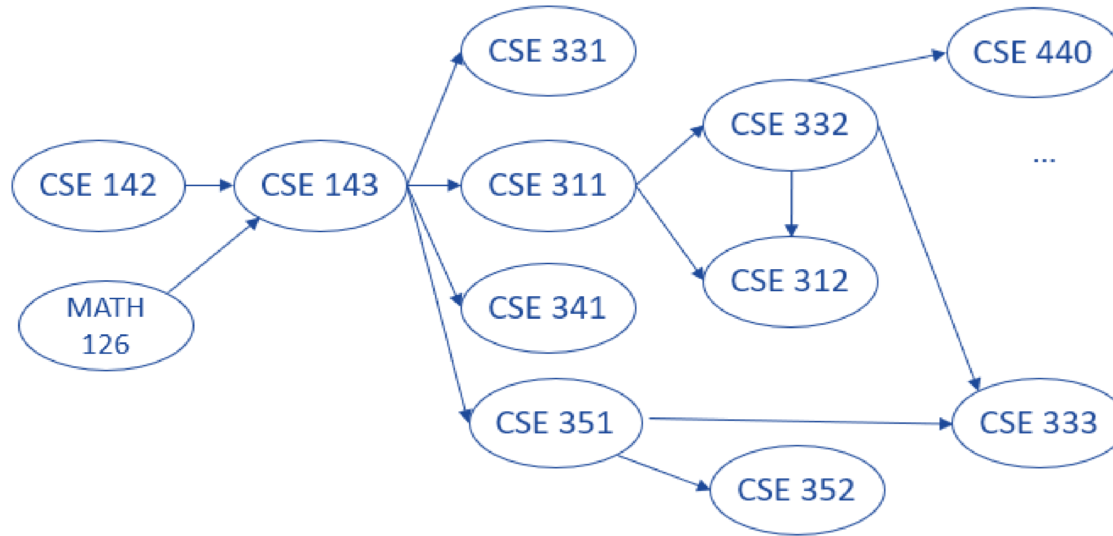
Output:

126 352
142
143
311
331
332
312
341
351
333

Node:	126	142	143	311	312	331	332	333	341	351	352	440
Removed?	x	x	x	x	x	x	x	x	x	x	x	
In-deg:	0	0	2	1	2	1	1	2	1	1	1	1
			1	0	1	0	0	1	0	0	0	0
			0		0			0				

الفرز الطوبولوجي

- مثال (1):



Output:

126
142
143
311
331
332
312
341
351
333
352
440

Node:	126	142	143	311	312	331	332	333	341	351	352	440
Removed?	x	x	x	x	x	x	x	x	x	x	x	x
In-deg:	0	0	2	1	2	1	1	2	1	1	1	1
			1	0	1	0	0	1	0	0	0	0
			0		0			0				

الفرز الطوبولوجي

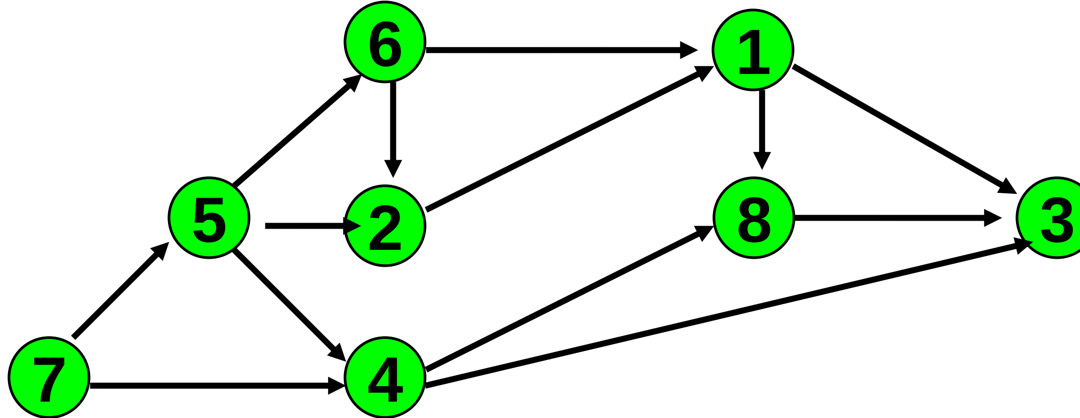
- مثال (2):

Initialization

Determine the indegree of each node

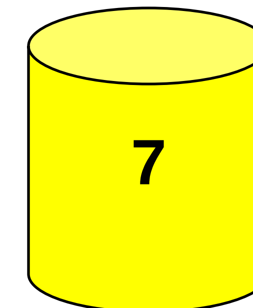
LIST is the set of nodes with indegree of 0.

“Next” will be the label of nodes in the topological order.



next 0

Node	1	2	3	4	5	6	7	8
Indegree	2	2	3	2	1	1	0	2



LIST

الفرز الطوبولوجي

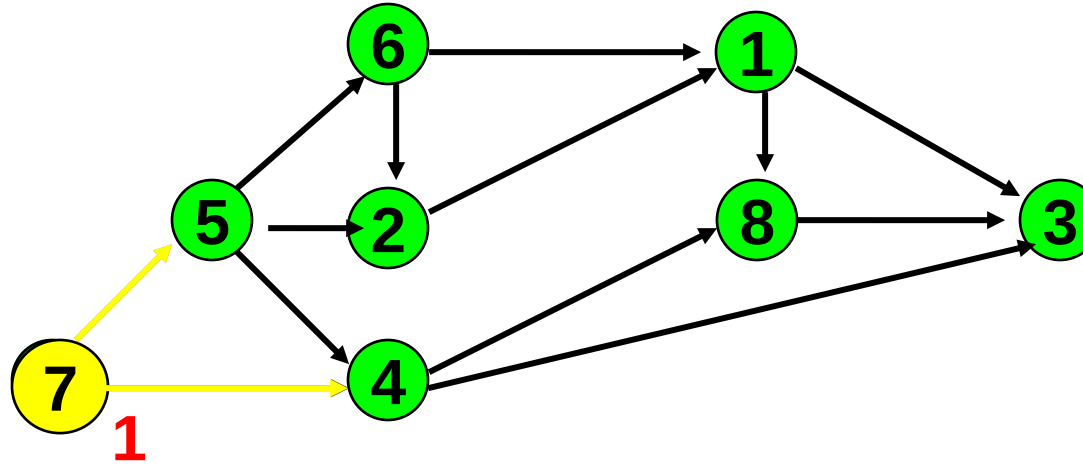
- مثال (2):

Select a node from LIST and delete it.

next := next +1
order(i) := next;

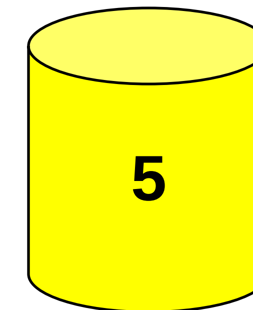
update indegrees

update LIST



next 1

Node	1	2	3	4	5	6		8
Indegree	2	2	3	1	0	1		2



LIST

الفرز الطوبولوجي

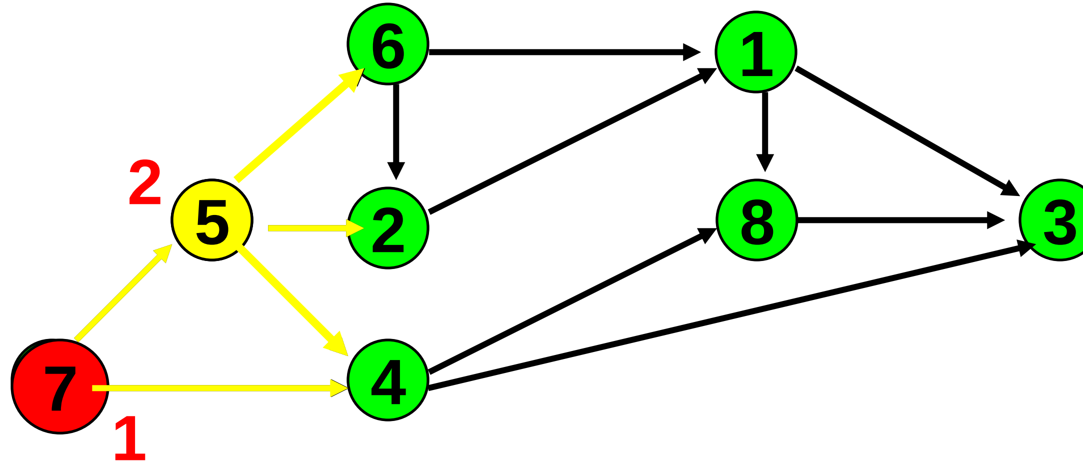
- مثال (2):

Select a node from LIST and delete it.

next := next +1
order(i) := next;

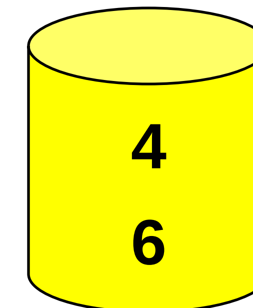
update indegrees

update LIST



next 2

Node	1	2	3	4		6		8
Indegree	2	1	3	0		0		2



LIST

الفرز الطوبولوجي

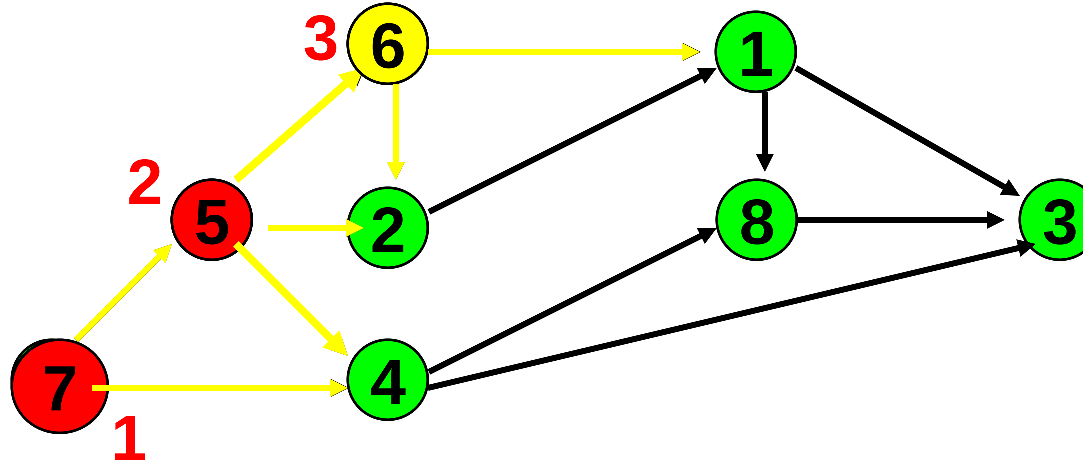
▪ مثال (2):

Select a node from LIST and delete it.

next := next +1
order(i) := next;

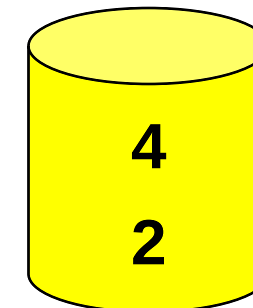
update indegrees

update LIST



next 3

Node	1	2	3	4	8
Indegree	1	0	3	0	2



LIST

الفرز الطوبولوجي

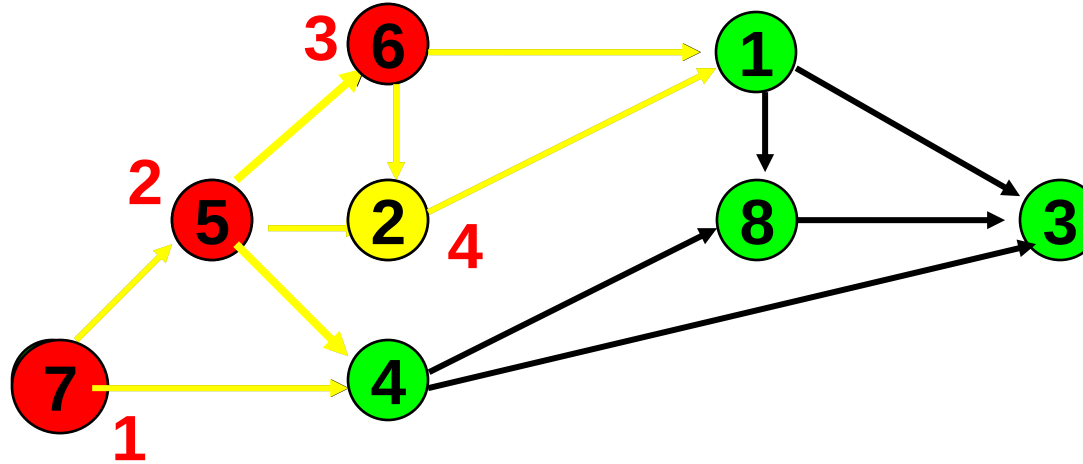
- مثال (2):

Select a node from LIST and delete it.

next := next +1
order(i) := next;

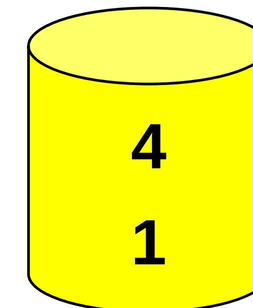
update indegrees

update LIST



next **4**

Node	1		3	4				8
Indegree	0		3	0				2



LIST

الفرز الطوبولوجي

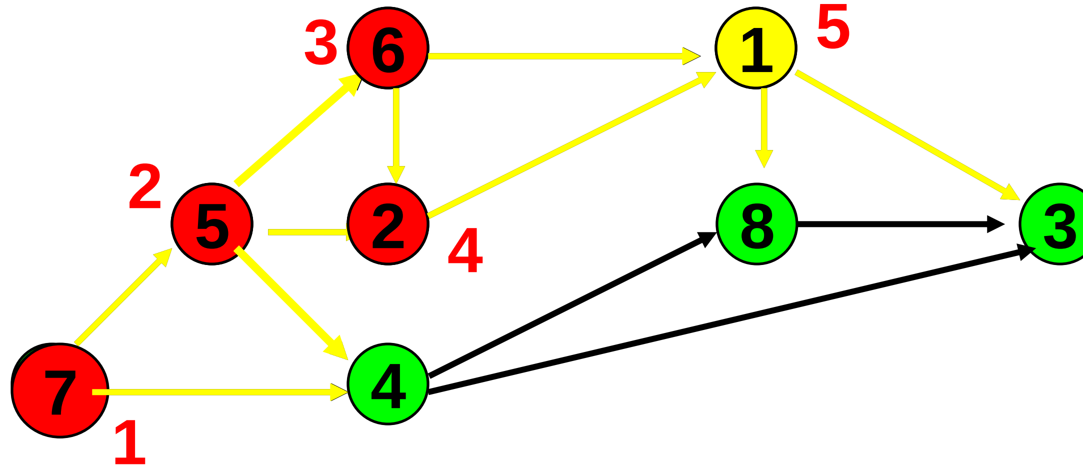
- مثال (2):

Select a node from LIST and delete it.

next := next +1
order(i) := next;

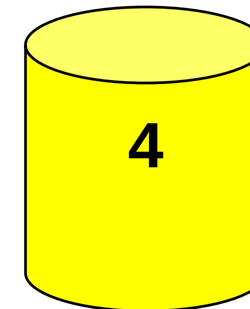
update indegrees

update LIST



next 5

Node			3	4			8
Indegree			2	0			1



LIST

الفرز الطوبولوجي

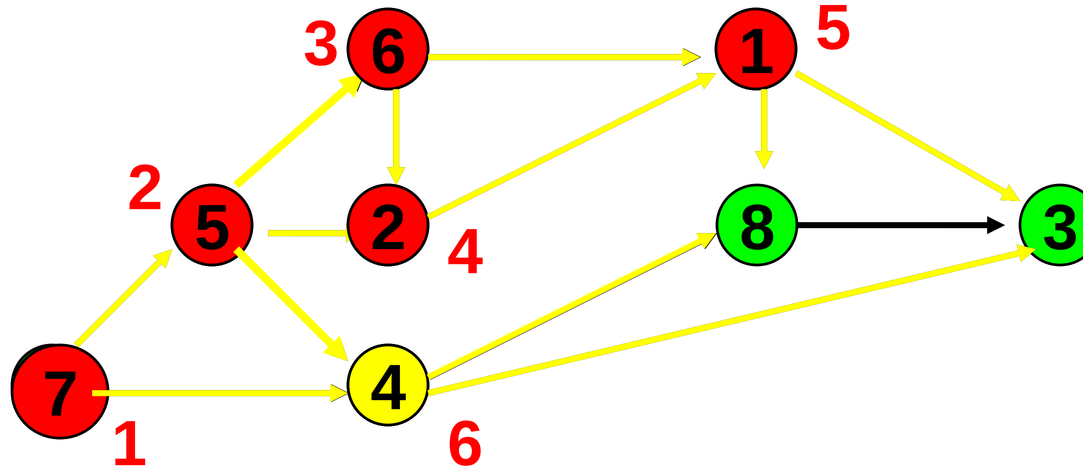
- مثال (2):

Select a node from LIST and delete it.

next := next +1
order(i) := next;

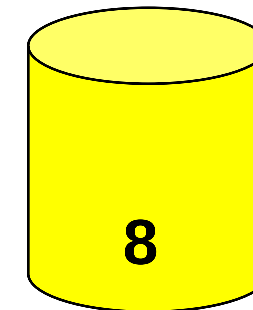
update indegrees

update LIST



next 6

Node	3	8
Indegree	1	0



LIST

الفرز الطوبولوجي

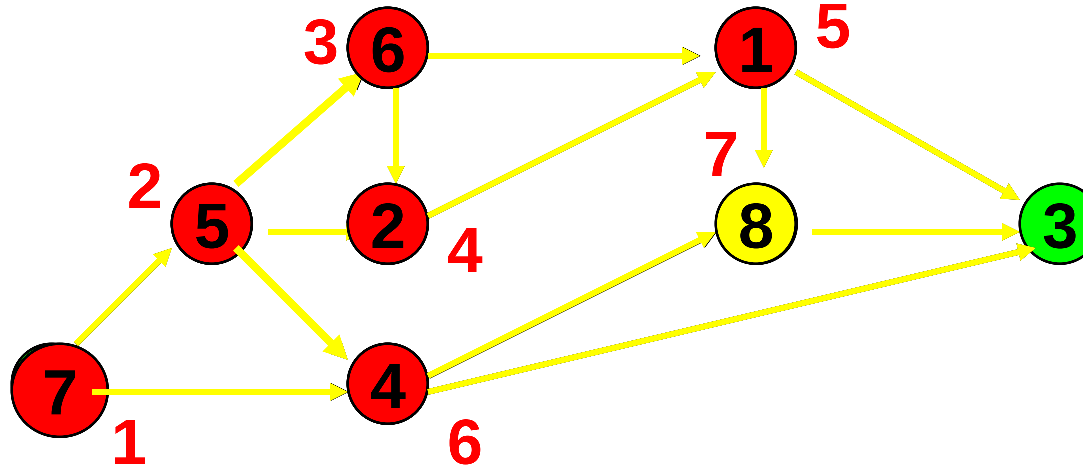
- مثال (2):

Select a node from LIST and delete it.

next := next +1
order(i) := next;

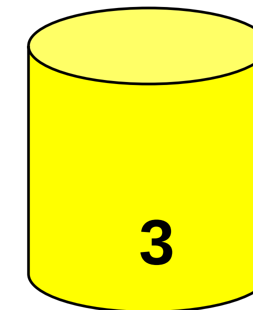
update indegrees

update LIST



next 7

Node		3
Indegree		0



LIST

الفرز الطوبولوجي

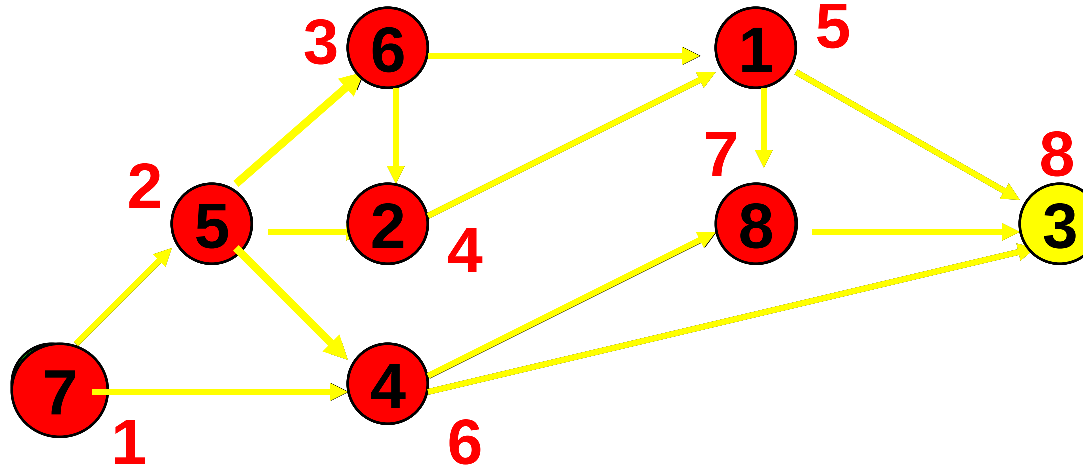
- مثال (2):

Select a node from LIST and delete it.

next := next +1
order(i) := next;

update indegrees

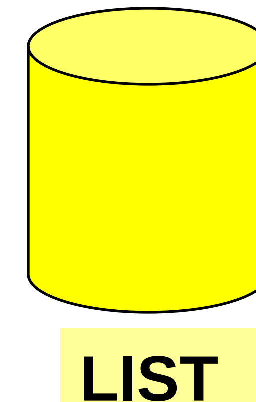
update LIST



next 8

Node	
Indegree	

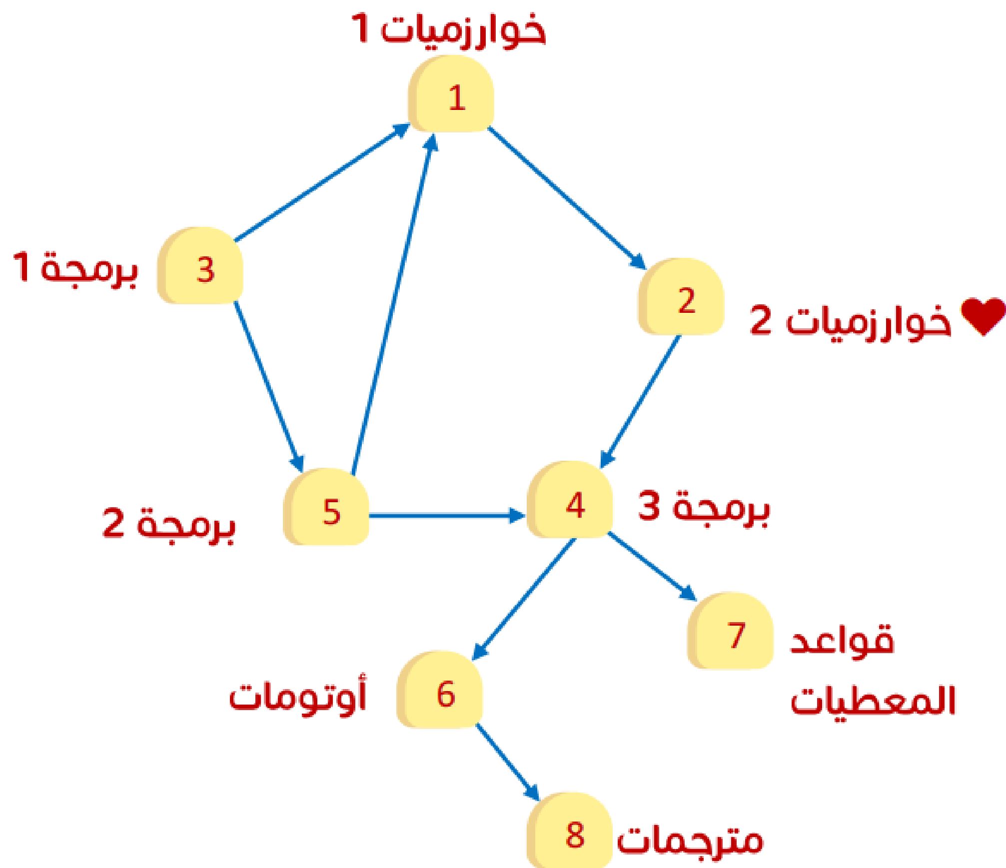
List is empty.The algorithm terminates with a topological order of the nodes



LIST

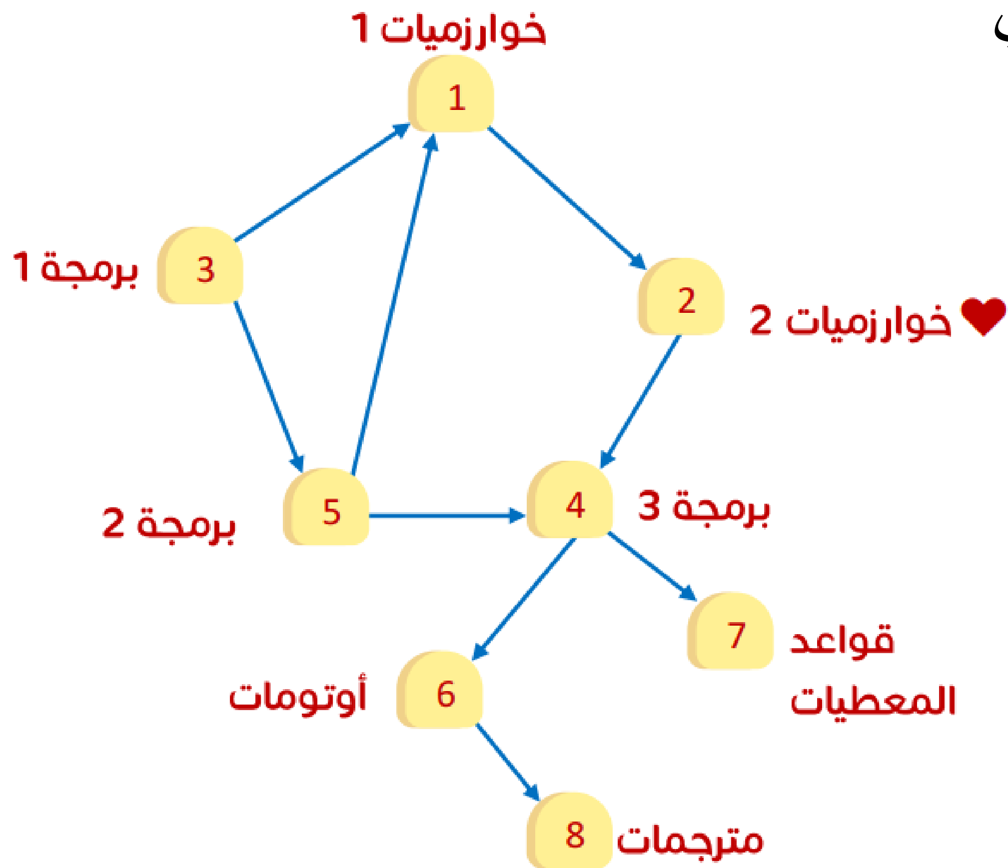
الفرز الطوبولوجي

- **مثال (3):** في البيان الموضّح جانباً كل عقدة من العقد تمثّل مادة معينة، والوصلة $u \to v$ تعني أن المادة v تأتي بعد المادة u (تعتمد عليها).
  - نريد إيجاد ترتيب لعقد البيان بحيث يكون الترتيب **موافقا لأولوية** المواد (مثلاً ليس من الممكن أن تبدأ بمقرر الخوارزميات (1) قبل أن تأخذ مقرر البرمجة(1+2) وهكذا).



الفرز الطوبولوجي

- **مثال (3):** لدينا أكثر من حل ويمكننا بأحدها فقط كالتالي



حل (1): برمجة 1 ← برمجة 2 ← خوارزميّات 1 ← خوارزميّات 2 ← برمجة 3 ← قواعد المعطيات ← أوتومات ← مترجمات.

حل (2): برمجة 1 ← برمجة 2 ← خوارزميّات 1 ← خوارزميّات 2 ← برمجة 3 ← أوتومات ← قواعد المعطيات ← مترجمات.

حل (3): برمجة 1 ← برمجة 2 ← خوارزميّات 1 ← خوارزميّات 2 ← برمجة 3 ← أوتومات ← مترجمات ← قواعد المعطيات.

مسألة المسار الأقصر

(shortest path problem)

مسألة المسار الأقصر

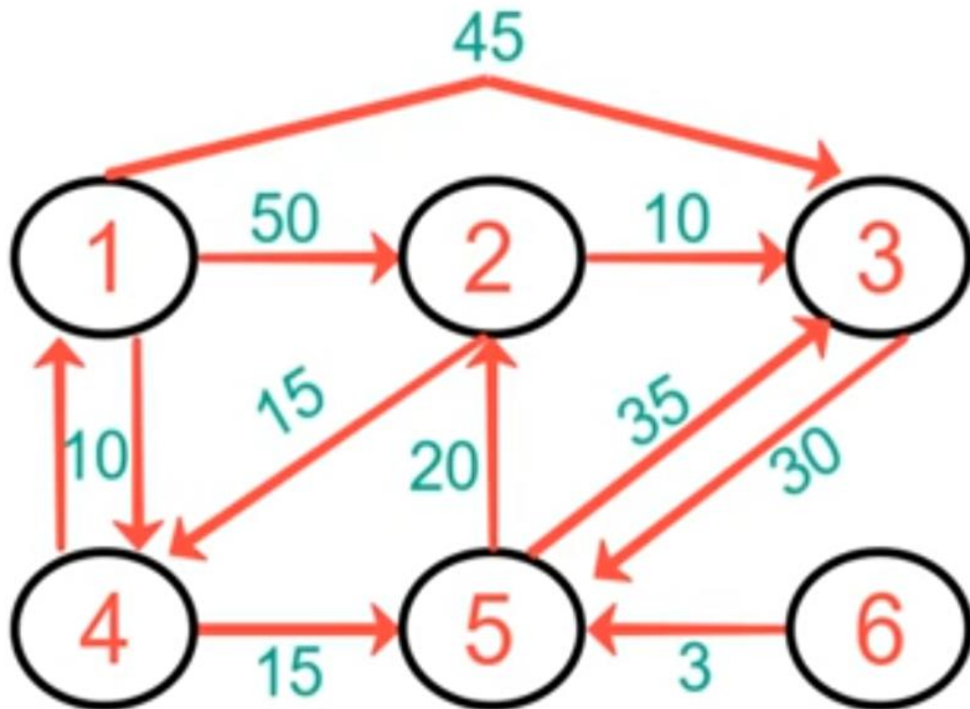
- لإيجاد أقصر طريق في **البيان غير الموزون** نستخدم خوارزمية BFS

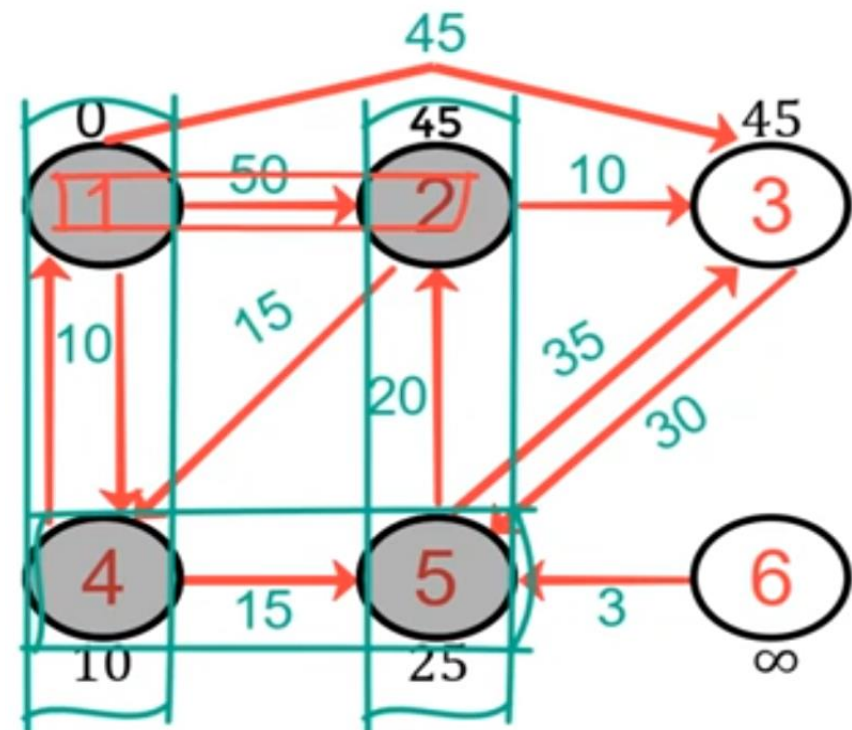
- لإيجاد أقصر طريق في **البيان الموزون** نستخدم خوارزميّة DIJKSTRA.
  - قد يدل الوزن على وقت، مسافة، كلفة مالية، .... حسب المسألة المدروسة.
  - يجب الانتباه إلى أن **طول المسار** في البيان الموزون هو **مجموع أوزان أضلاع** المسار وليس عدد الأضلاع.

- نميز بين:
  - Single-Source Shortest Paths (SSSP)
  - All Pairs Shortest Paths

مسألة المسار الأقصر

- في مسألة SSSP، يتم تحديد رأس بداية اختياري:
  1. Single Source one Destination: إيجاد **اقصر مسار** من رأس البداية إلى الرأس الهدف
     - **مثال:** المسار الاقصر من الرأس 1 إلى الرأس 2 هو المسار {1,4,5,2} وطوله يساوي 45=10+15+20 بينما باستخدام المسار المباشر تكون الكلفة أعلى وتساوي 50.

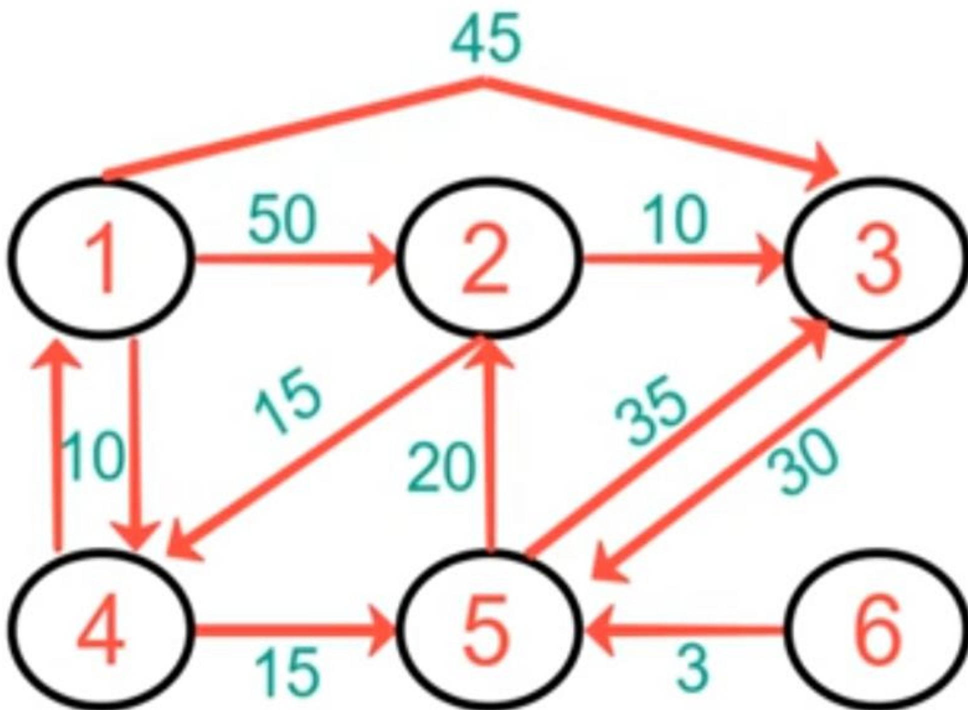




مسألة المسار الأقصر

- في مسألة SSSP، يتم تحديد رأس بداية اختياري:

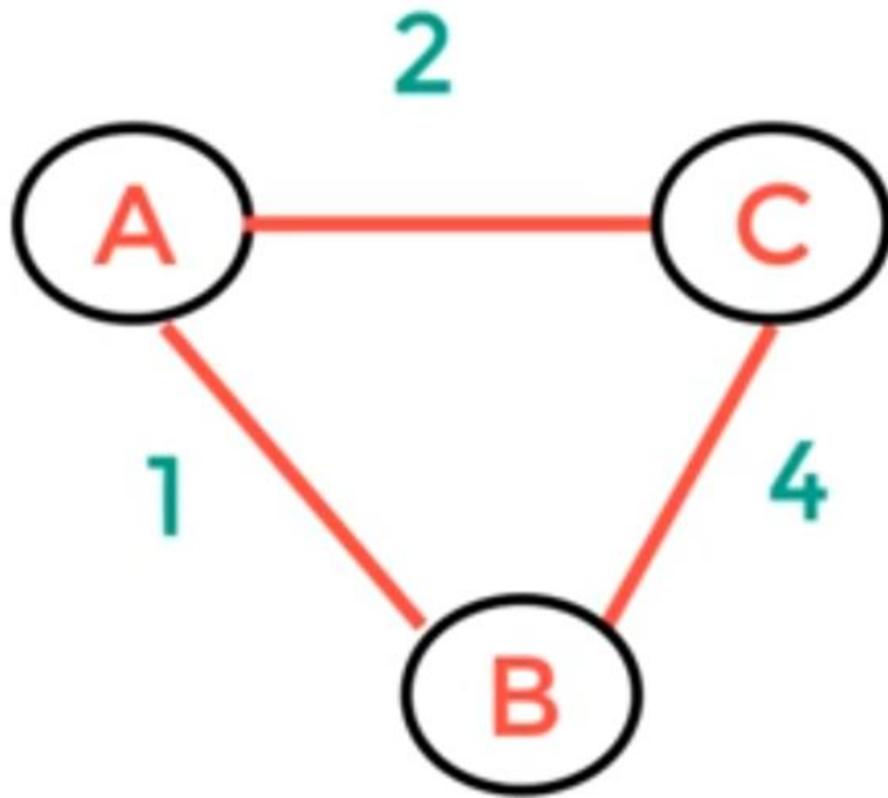
2. Single Source All Destinations: إيجاد **المسارات الأقصر** من رأس البداية إلى بقية رؤوس البيان.
   - مثال: المسارات الاقصر من الرأس 1

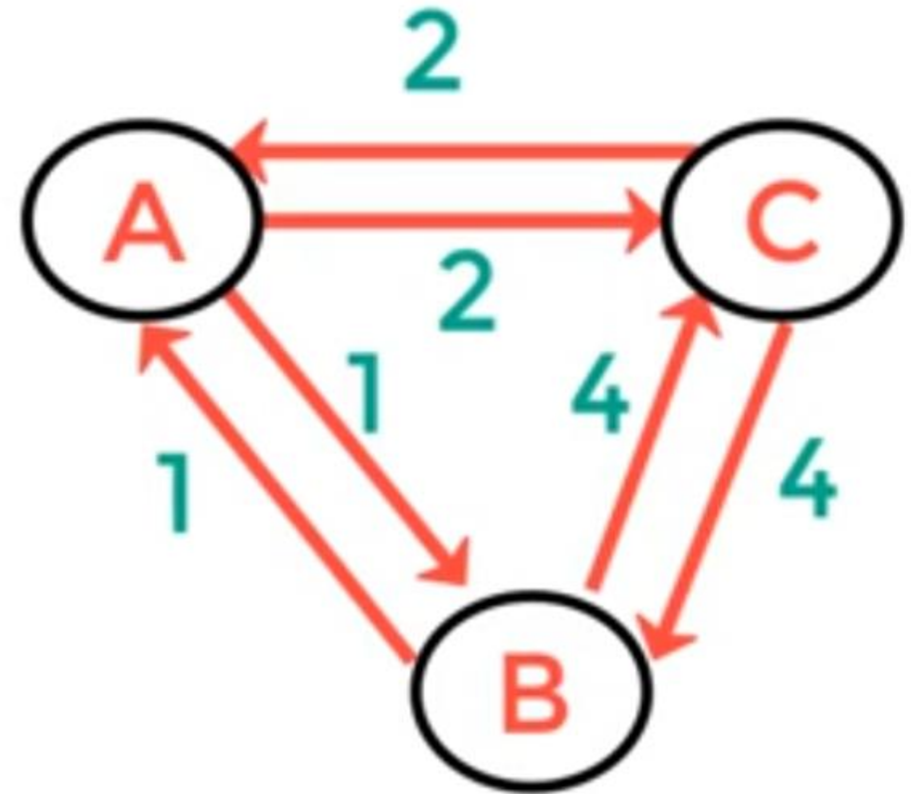


Path	length
{1,4}	10
{1,4,5}	25
{1,4,5,2}	45
{1,3}	45

مسألة المسار الأقصر

- ملاحظة: في حالة البيان غير الموجه نقوم بوضع الاوزان من أجل الاتجاهين.





Dijkstra's Algorithm

- نوضح خوارزمية DIJKSTRA في بيان موزون و رأس بداية s على الشكل التالي :

$d[s] \leftarrow 0$ (distance to source vertex is zero)

for each $v \in V - \{s\}$ (set all other distances to infinity)

$\quad d[v] \leftarrow \infty$

$S \leftarrow \varnothing$ (S, the set of visited vertices , initially empty)

$Q \leftarrow V$ (Q, a priority queue maintaining V – S, initially contains all vertices)

while (Q *is not empty*) **do**

$\quad u \leftarrow Q.dequeue()$ (select the element of Q with the min. distance)

$\quad S \leftarrow S \cup \{u\}$ (add u to list of visited vertices)

$\quad$ **for** each $v \in Adj[u]$

relaxation step

$\quad\quad$ **if** $d[v] > d[u] + w(u, v)$ (if new shortest path found)

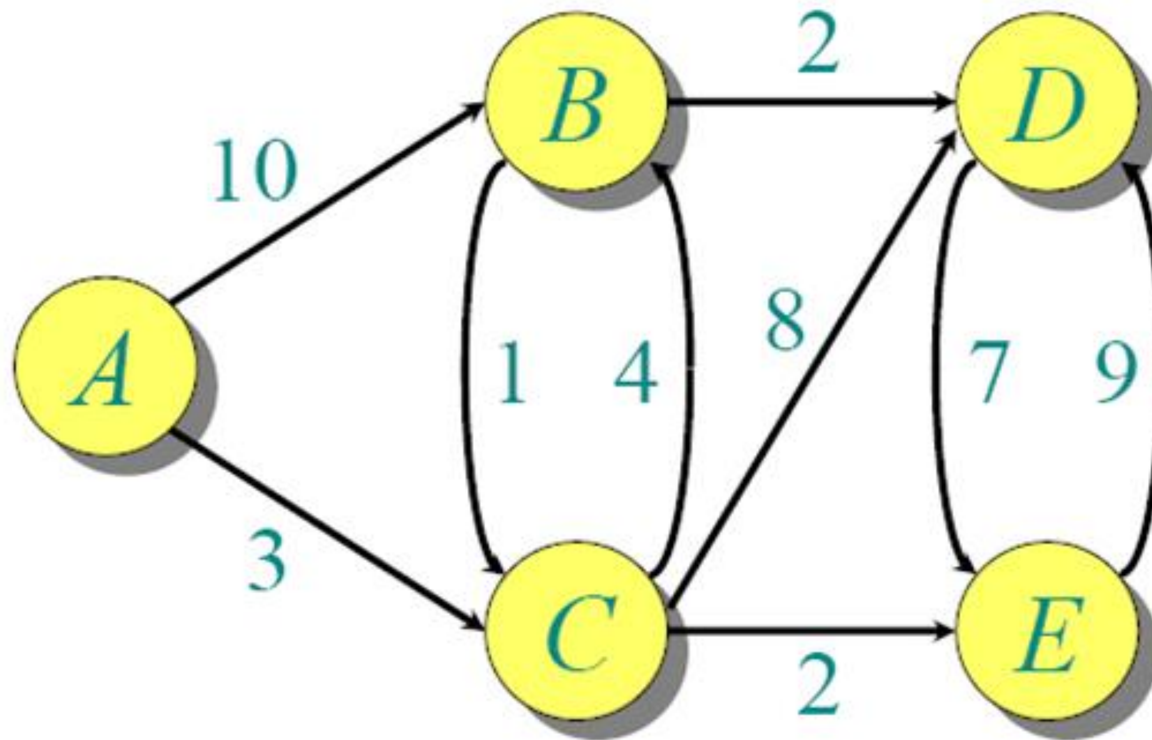
$\quad\quad\quad$ **then** $d[v] \leftarrow d[u] + w(u, v)$ (set new value of shortest path)

$\quad$ (if desired, add traceback code: predecessor(v)=u;)

return d;

Example of Dijkstra's algorithm

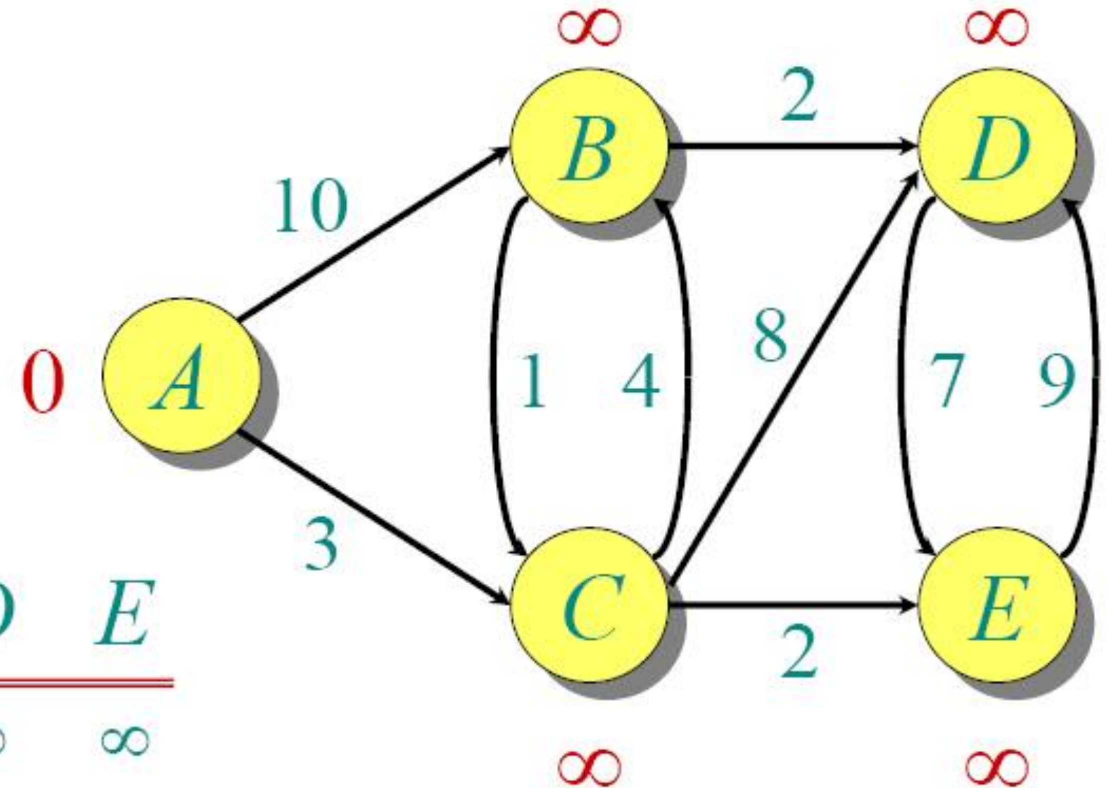
- مثال (1):



source vertex: A

Example of Dijkstra's algorithm

Initialize:



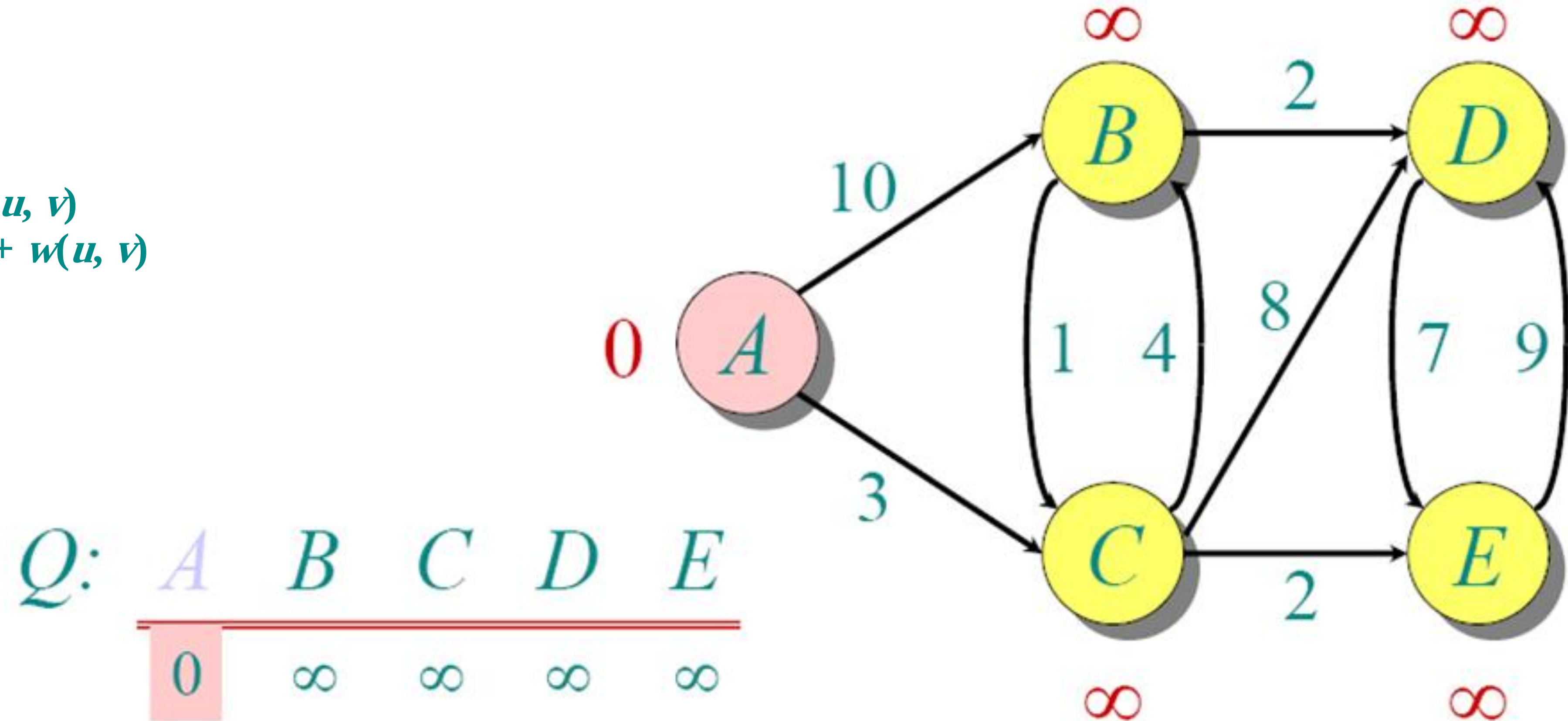
Q:	A	B	C	D	E
	0	∞	∞	∞	∞

S: {}

$d[s] \leftarrow 0$
for each $v \in V - \{s\}$
    $d[v] \leftarrow \infty$
$S \leftarrow \varnothing$
$Q \leftarrow V$

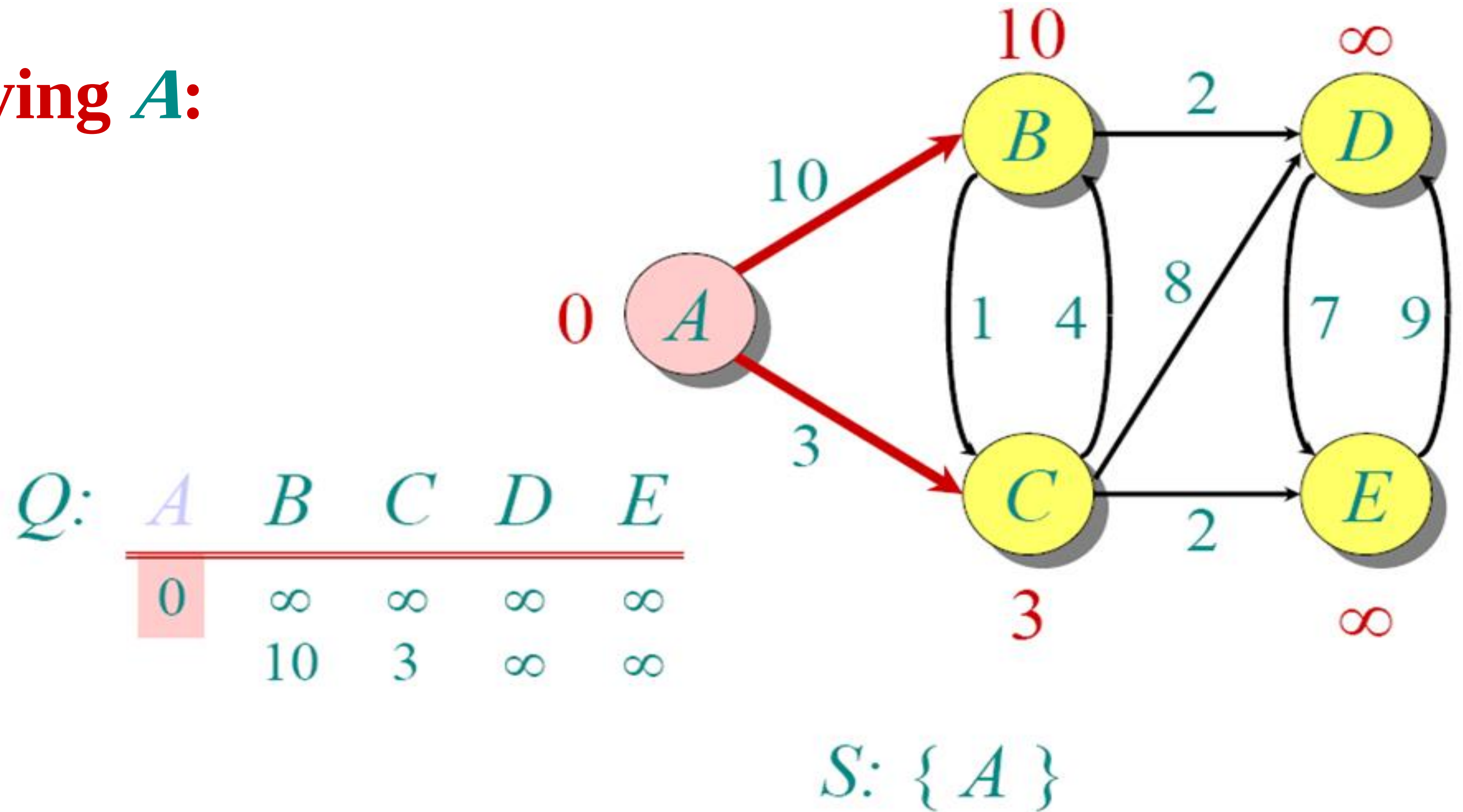
Example of Dijkstra's algorithm

while (*Q is not empty)* **do**
  $u \leftarrow Q.dequeue()$
  $S \leftarrow S \cup \{u\}$
  for each $v \in Adj[u]$
    if $d[v] > d[u] + w(u, v)$
    then $d[v] \leftarrow d[u] + w(u, v)$



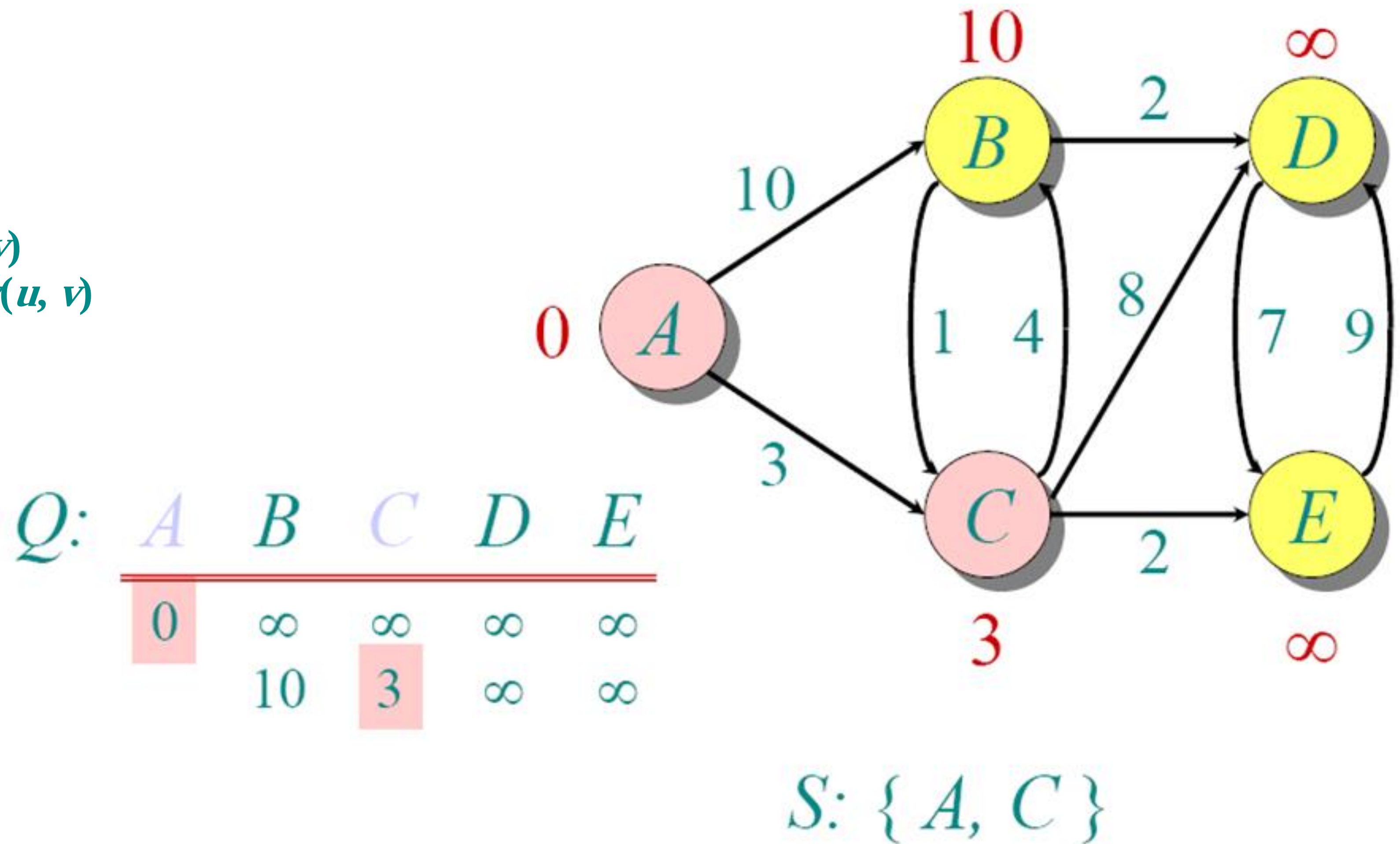
Example of Dijkstra's algorithm

Relax all edges leaving *A*:



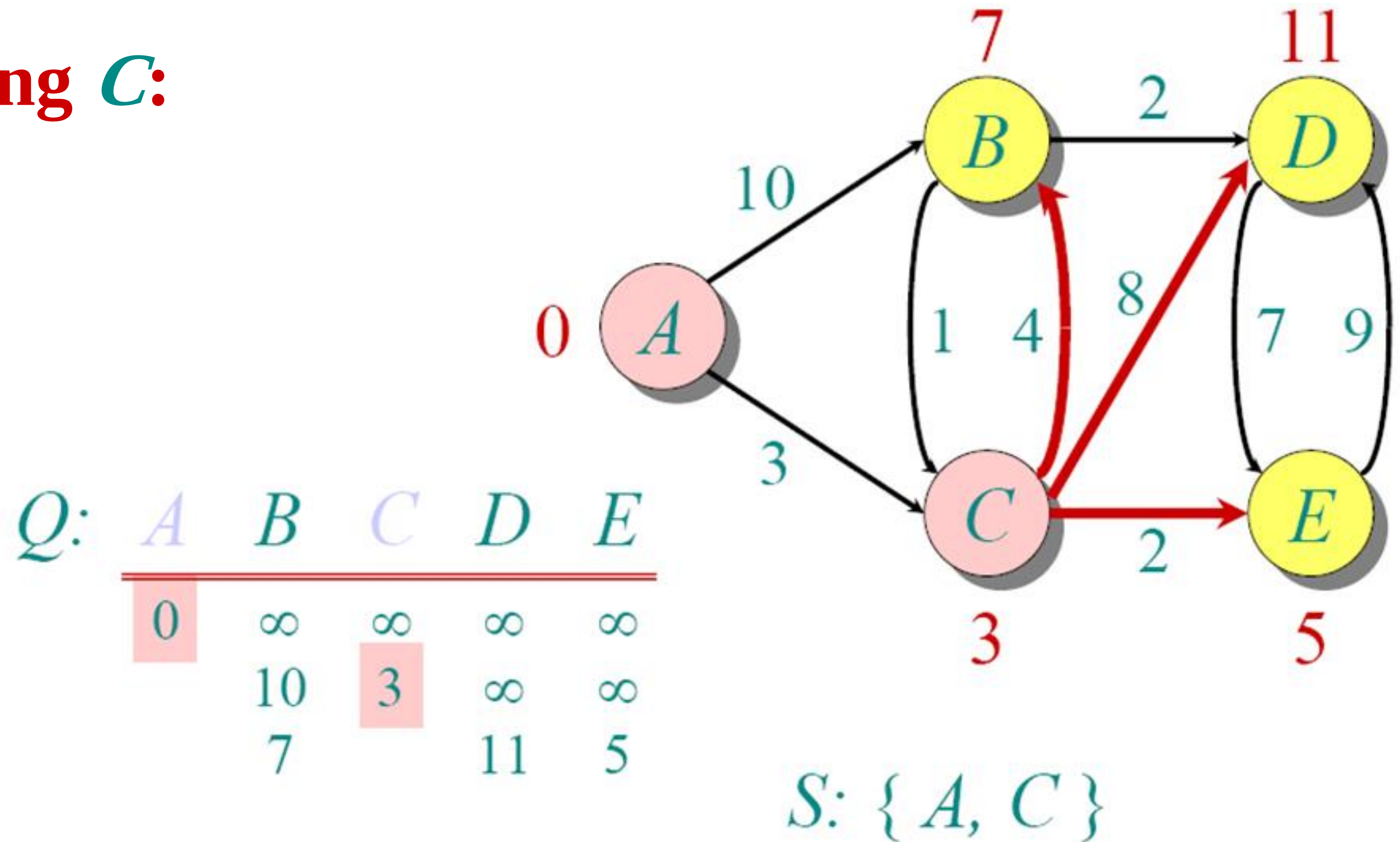
Example of Dijkstra's algorithm

while (*Q is not empty*) **do**
  $u \leftarrow Q.dequeue()$
  $S \leftarrow S \cup \{u\}$
  for each $v \in Adj[u]$
    if $d[v] > d[u] + w(u, v)$
    then $d[v] \leftarrow d[u] + w(u, v)$



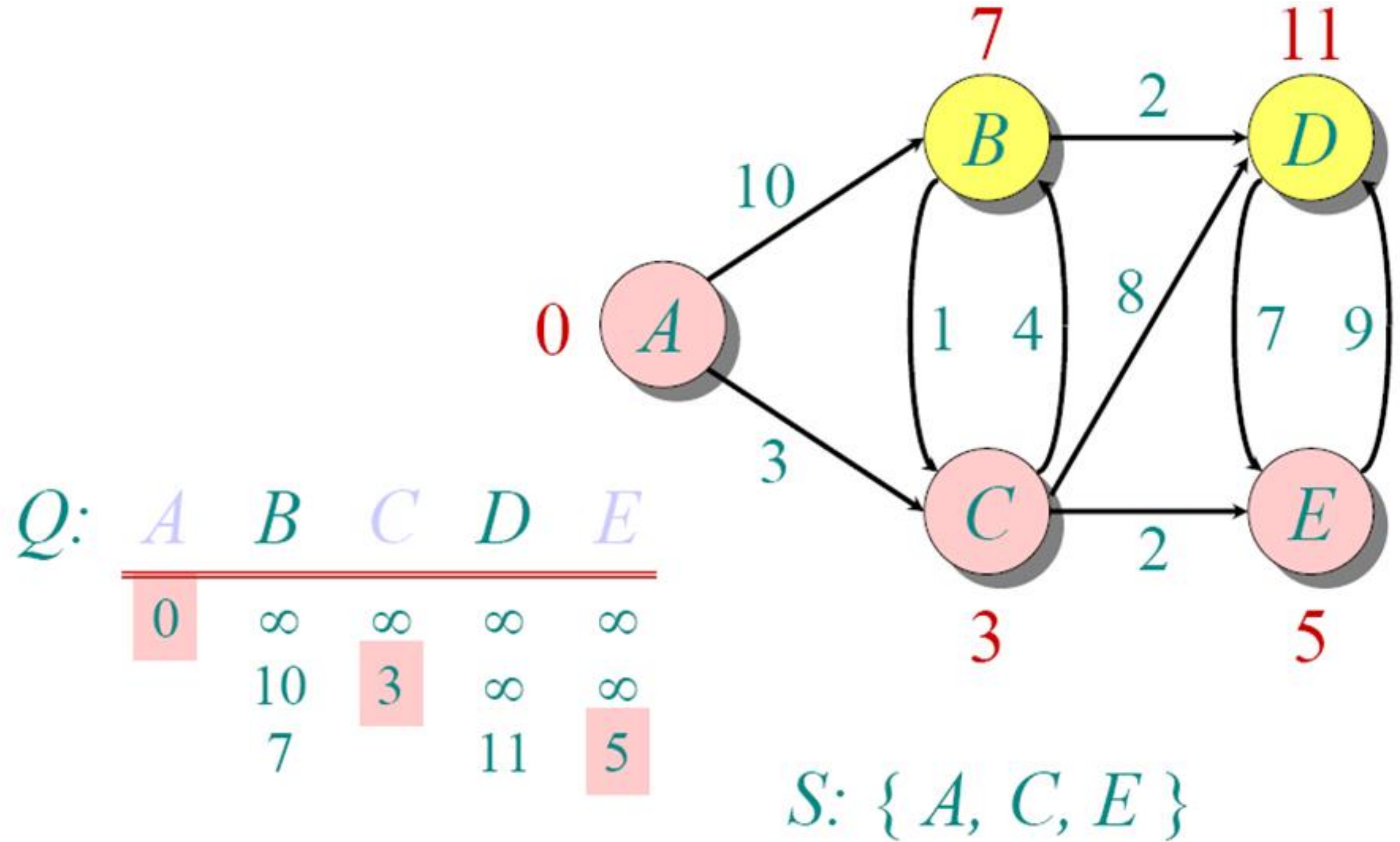
Example of Dijkstra's algorithm

Relax all edges leaving *C*:



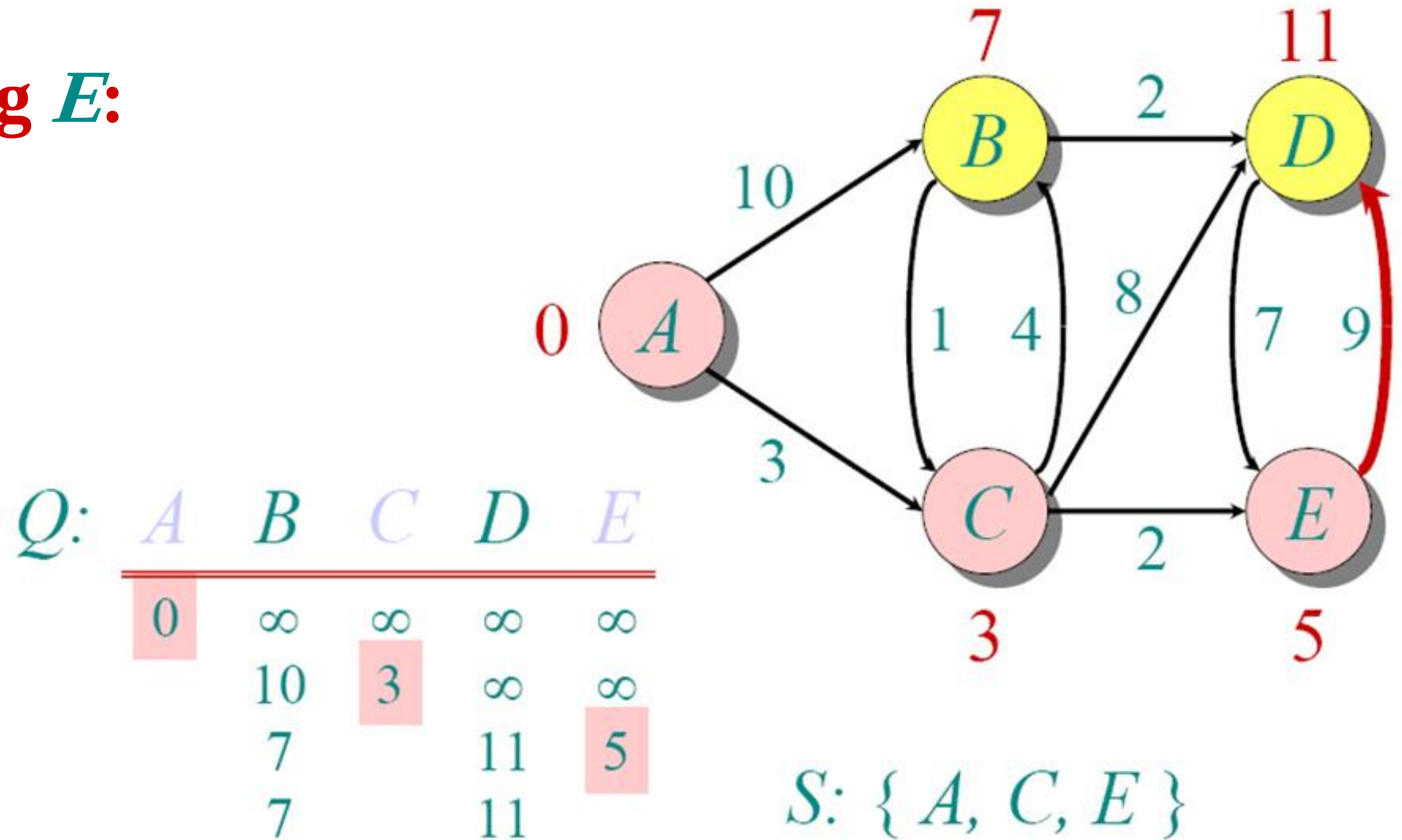
Example of Dijkstra's algorithm

while (*Q is not empty)* **do**
    $u \leftarrow Q.dequeue()$
    $S \leftarrow S \cup \{u\}$
    for each $v \in Adj[u]$
        if $d[v] > d[u] + w(u, v)$
        then $d[v] \leftarrow d[u] + w(u, v)$



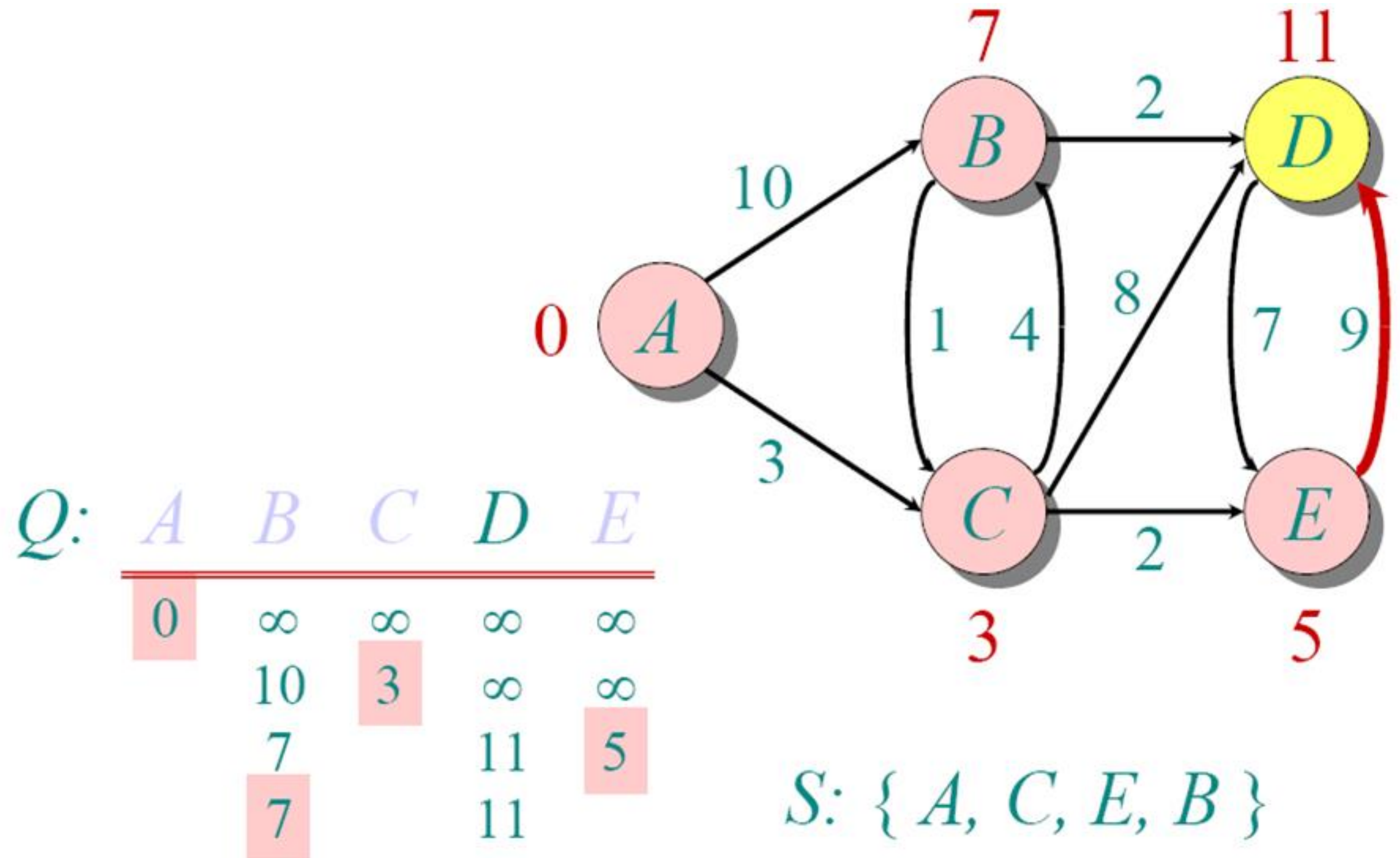
Example of Dijkstra's algorithm

Relax all edges leaving *E*:



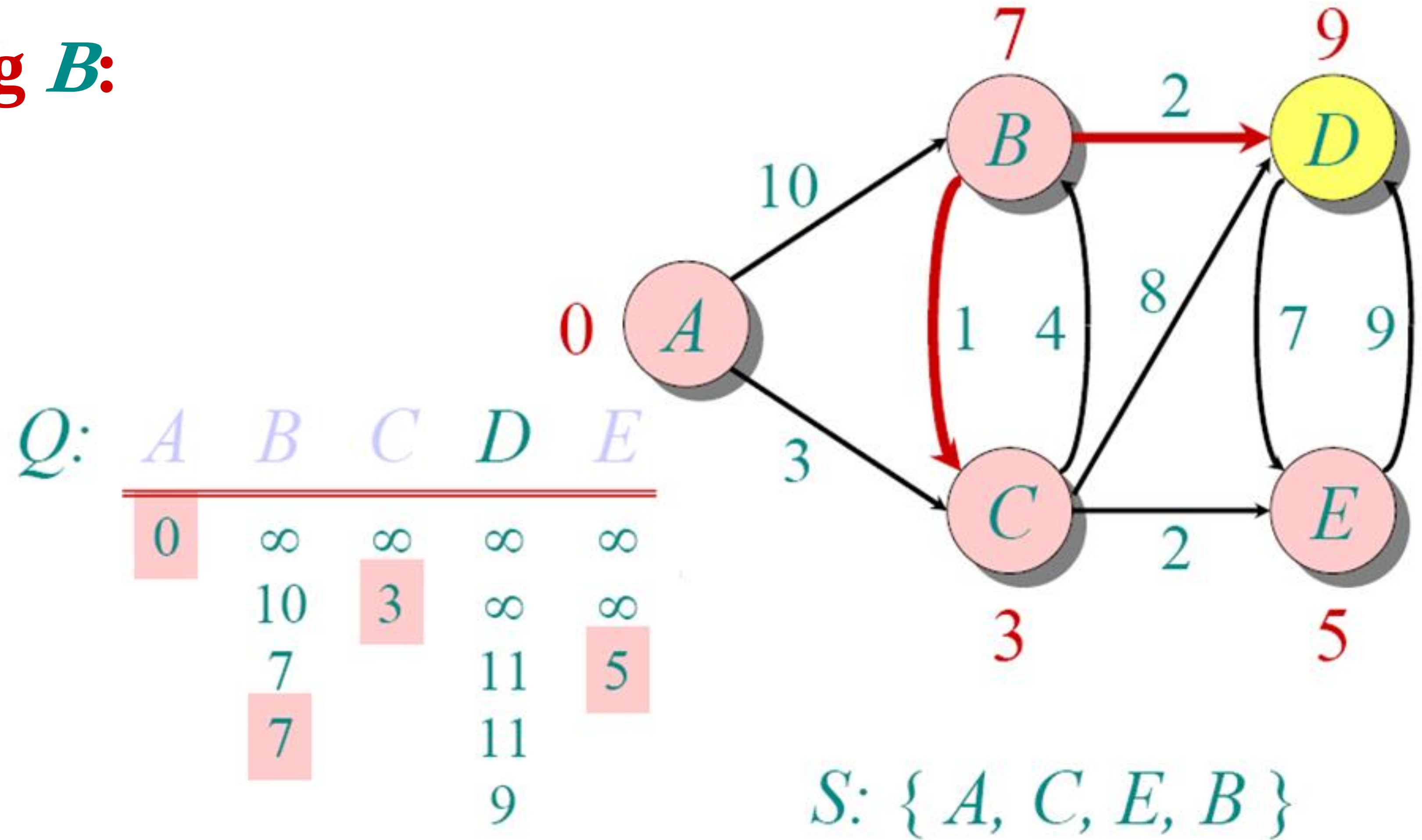
Example of Dijkstra's algorithm

while (*Q is not empty)* **do**
    $u \leftarrow Q.dequeue()$
    $S \leftarrow S \cup \{u\}$
    for each $v \in Adj[u]$
        if $d[v] > d[u] + w(u, v)$
        then $d[v] \leftarrow d[u] + w(u, v)$



Example of Dijkstra's algorithm

Relax all edges leaving *B*:



Example of Dijkstra's algorithm

while (*Q is not empty)* **do**

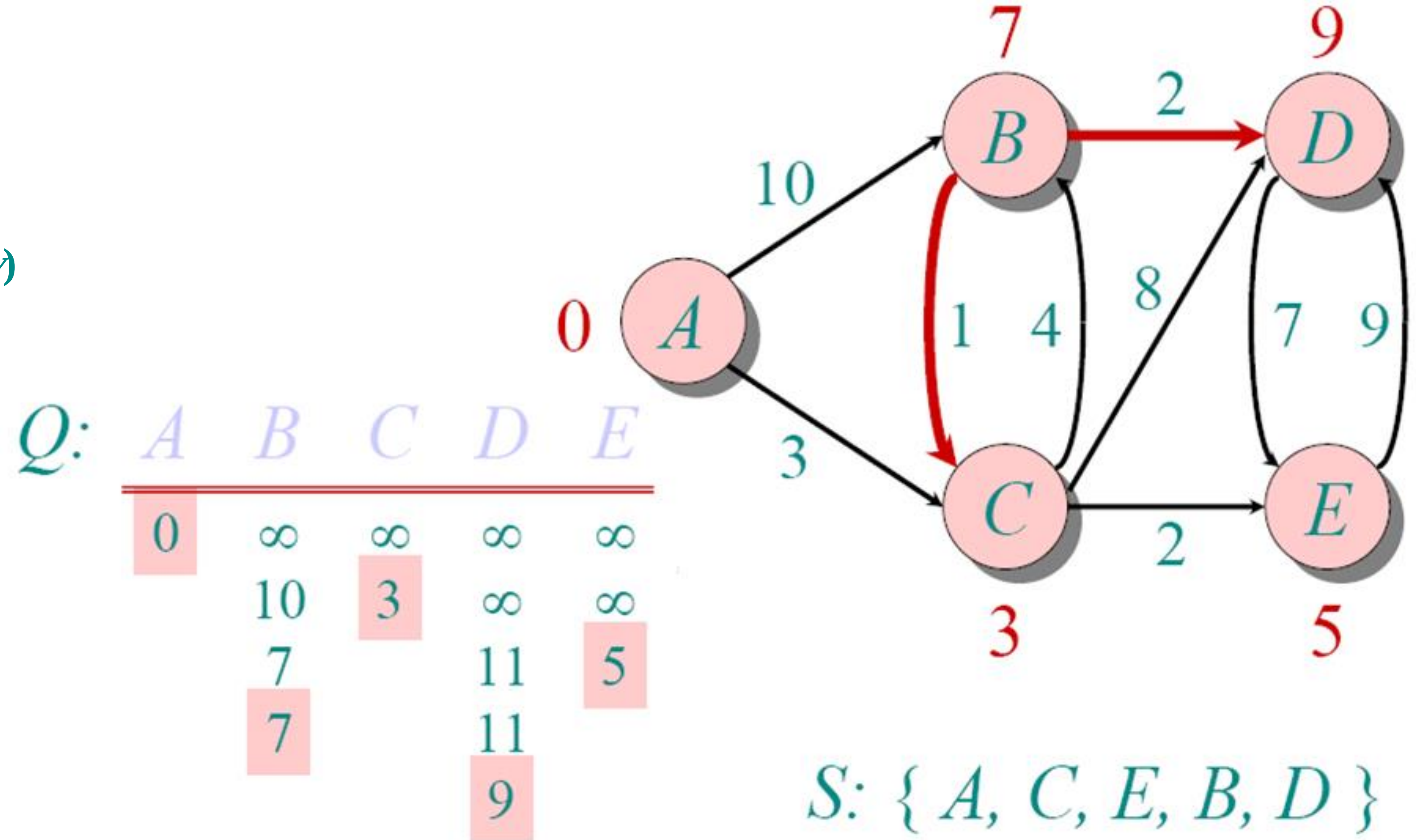
  $u \leftarrow Q.dequeue()$

  $S \leftarrow S \cup \{u\}$

  for each $v \in Adj[u]$

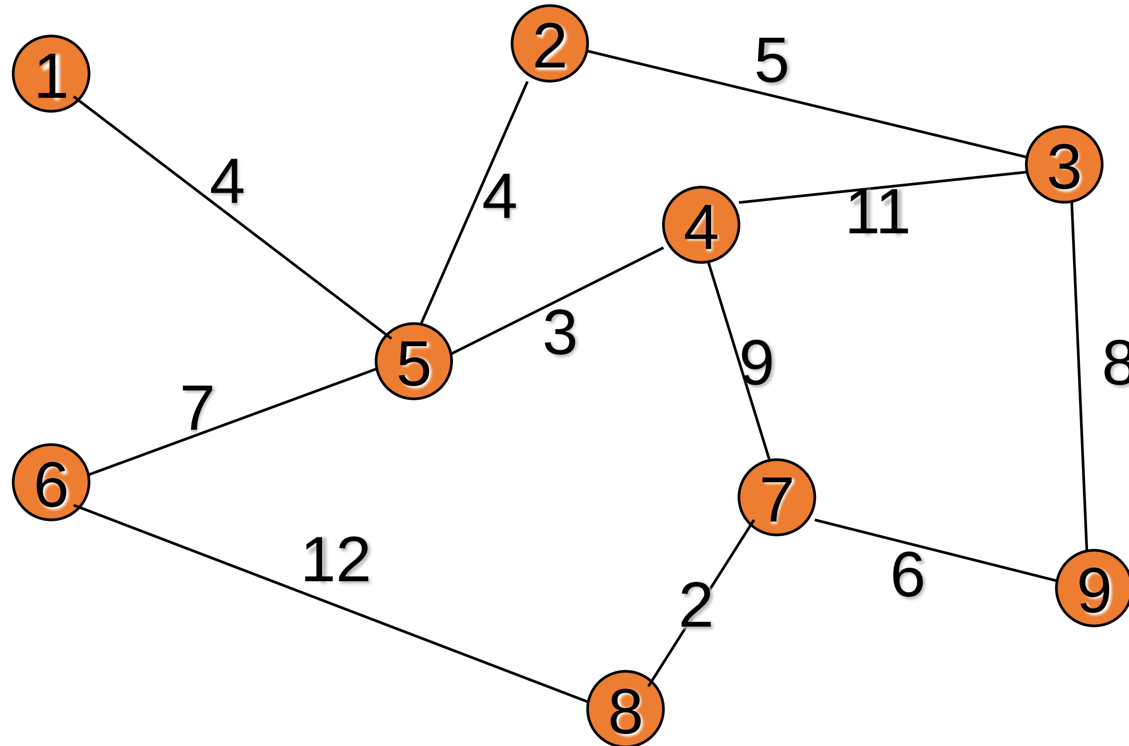
    if $d[v] > d[u] + w(u, v)$

    then $d[v] \leftarrow d[u] + w(u, v)$



Example of Dijkstra's algorithm

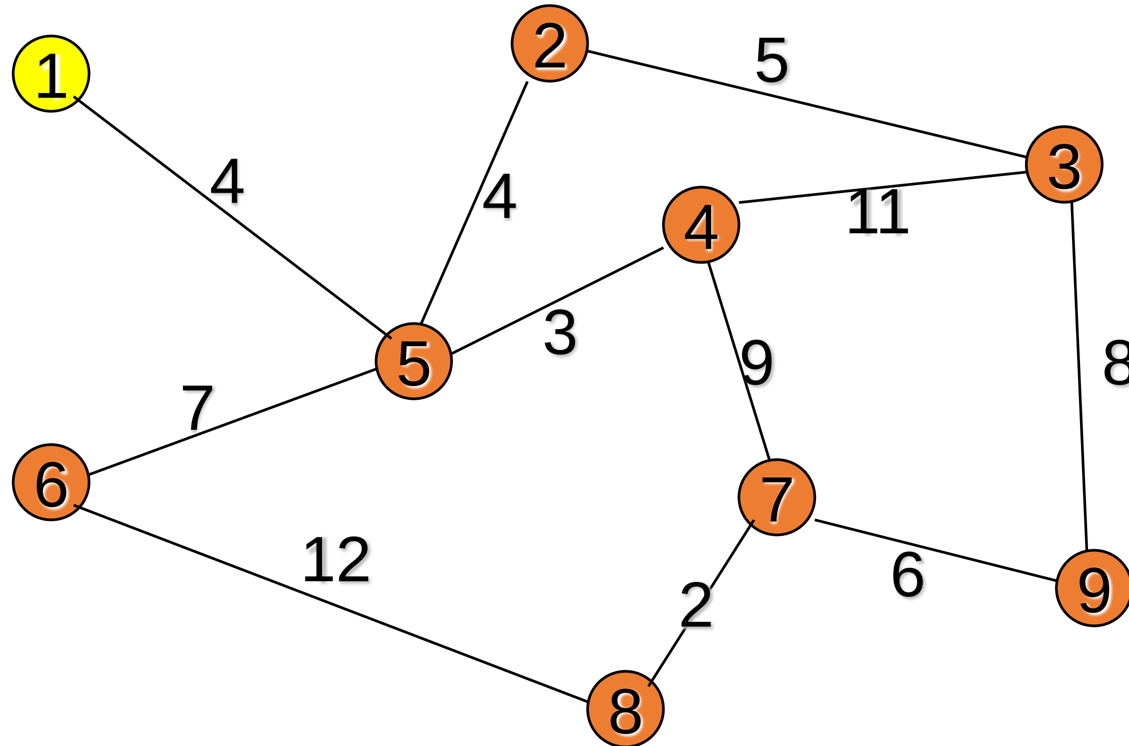
- مثال (2):



Processed	1	2	3	4	5	6	7	8	9
distance	0	?	?	?	?	?	?	?	?
predecessor	-1	-1	-1	-1	-1	-1	-1	-1	-1

Example of Dijkstra's algorithm

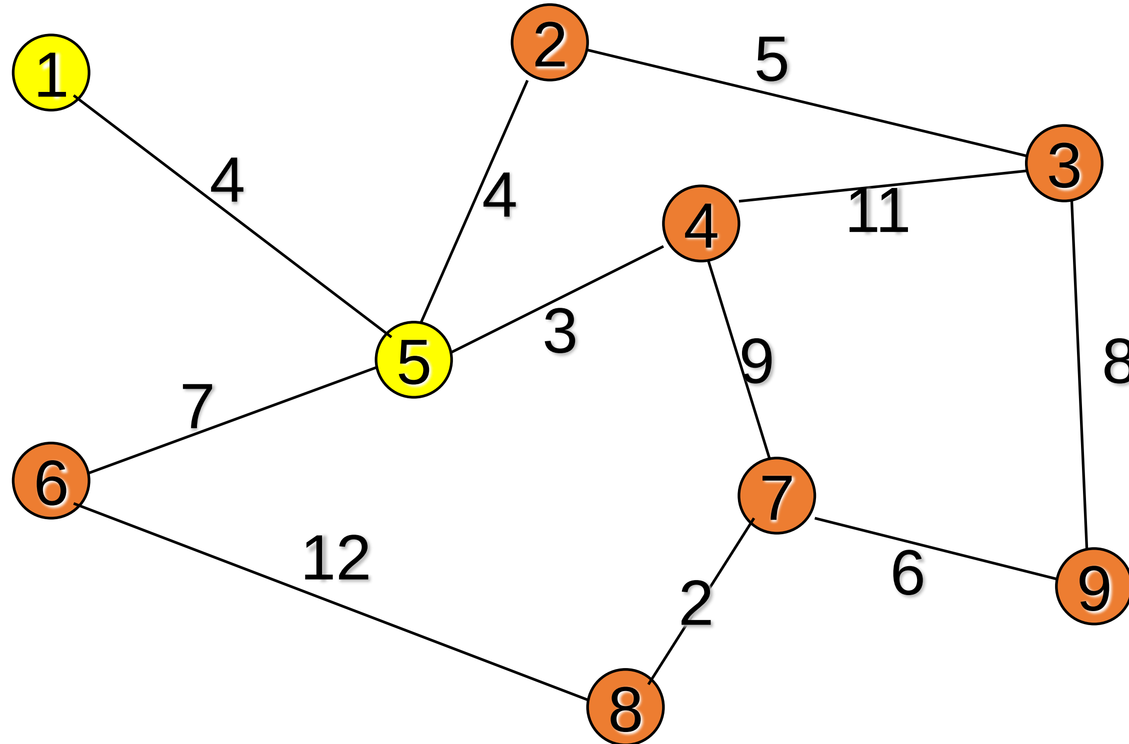
- مثال (2):



Processed	1	2	3	4	5	6	7	8	9
distance	0	?	?	?	4	?	?	?	?
predecessor	-1	-1	-1	-1	1	-1	-1	-1	-1

Example of Dijkstra's algorithm

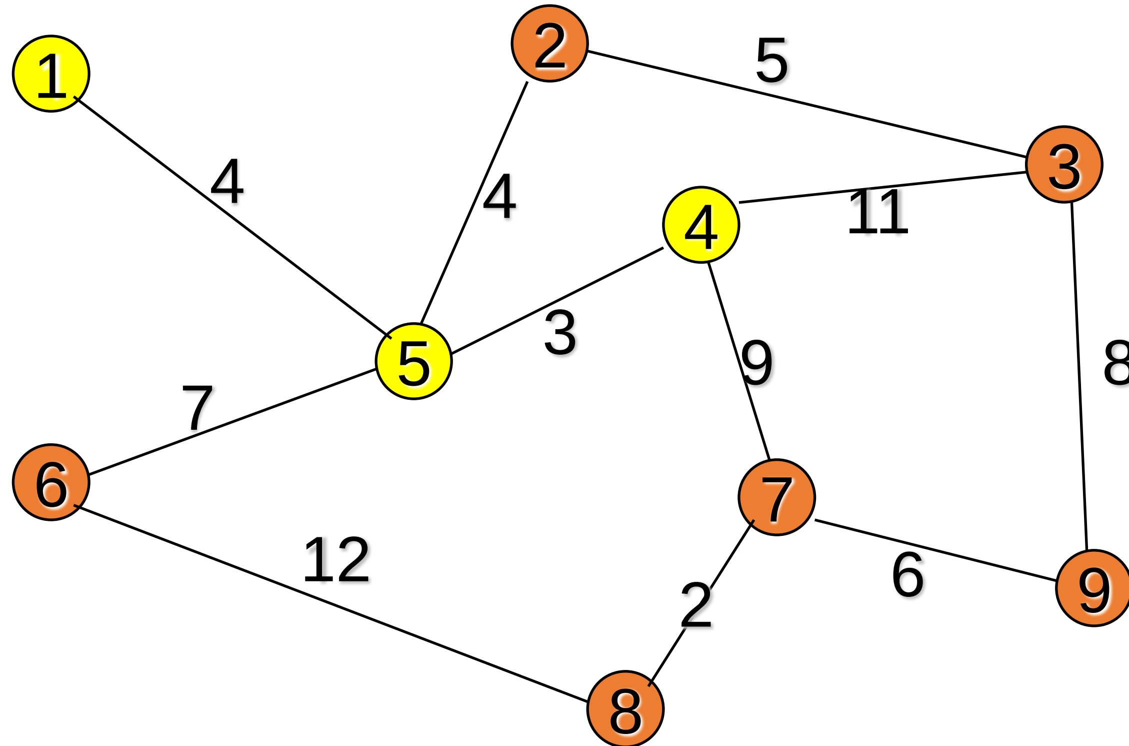
- مثال (2):



Processed	1	2	3	4	5	6	7	8	9
distance	0	8	?	7	4	11	?	?	?
predecessor	-1	5	-1	5	1	5	-1	-1	-1

Example of Dijkstra's algorithm

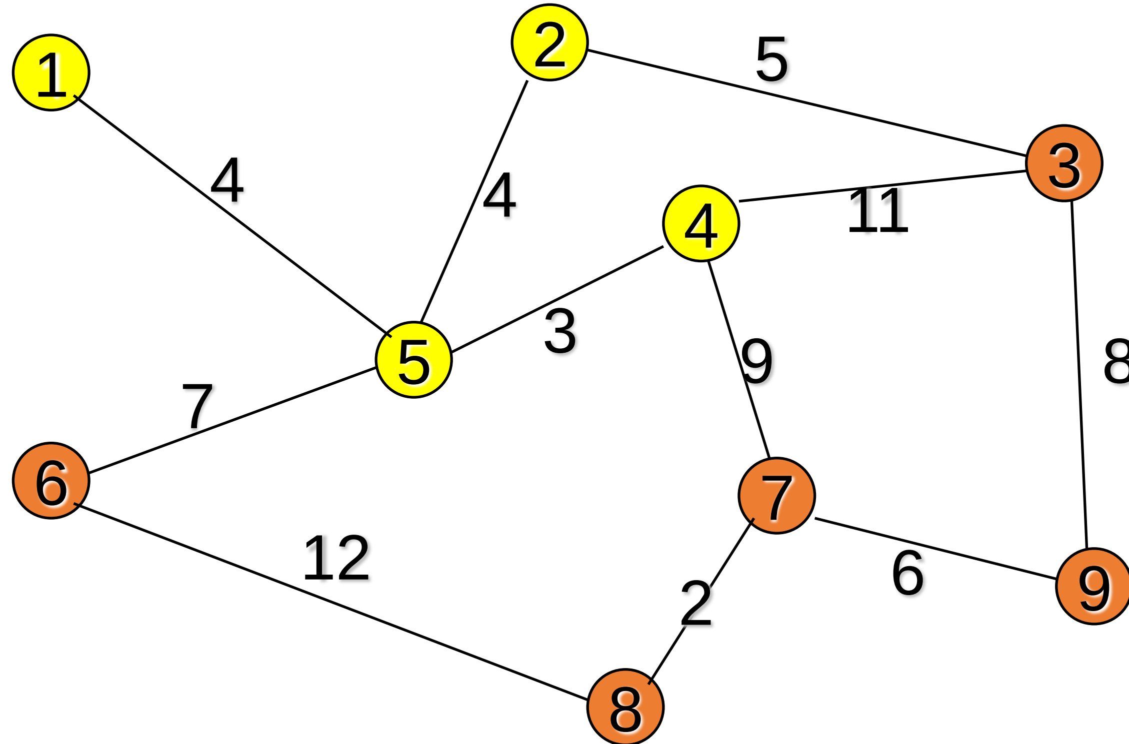
- مثال (2):



Processed	1	2	3	4	5	6	7	8	9
distance	0	8	18	7	4	11	16	?	?
predecessor	-1	5	4	5	1	5	4	-1	-1

Example of Dijkstra's algorithm

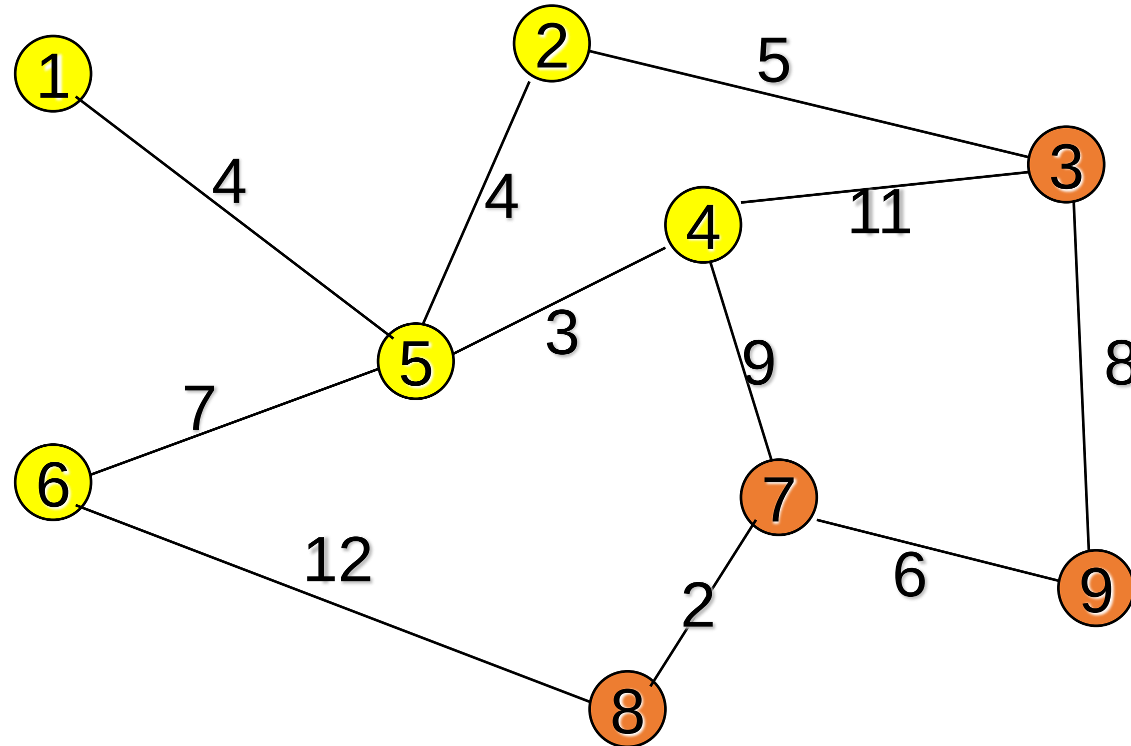
- مثال (2):



Processed	1	2	3	4	5	6	7	8	9
distance	0	8	13	7	4	11	16	?	?
predecessor	-1	5	2	5	1	5	4	-1	-1

Example of Dijkstra's algorithm

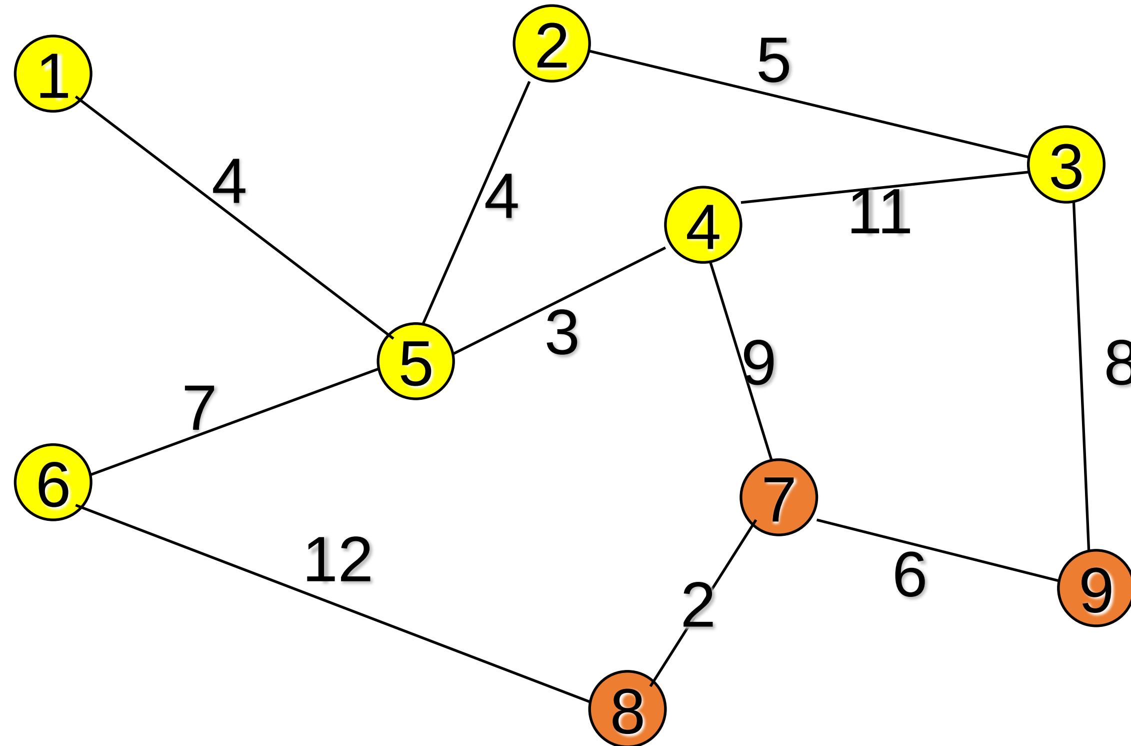
▪ مثال (2):



Processed	1	2	3	4	5	6	7	8	9
distance	0	8	13	7	4	11	16	23	?
predecessor	-1	5	2	5	1	5	4	6	-1

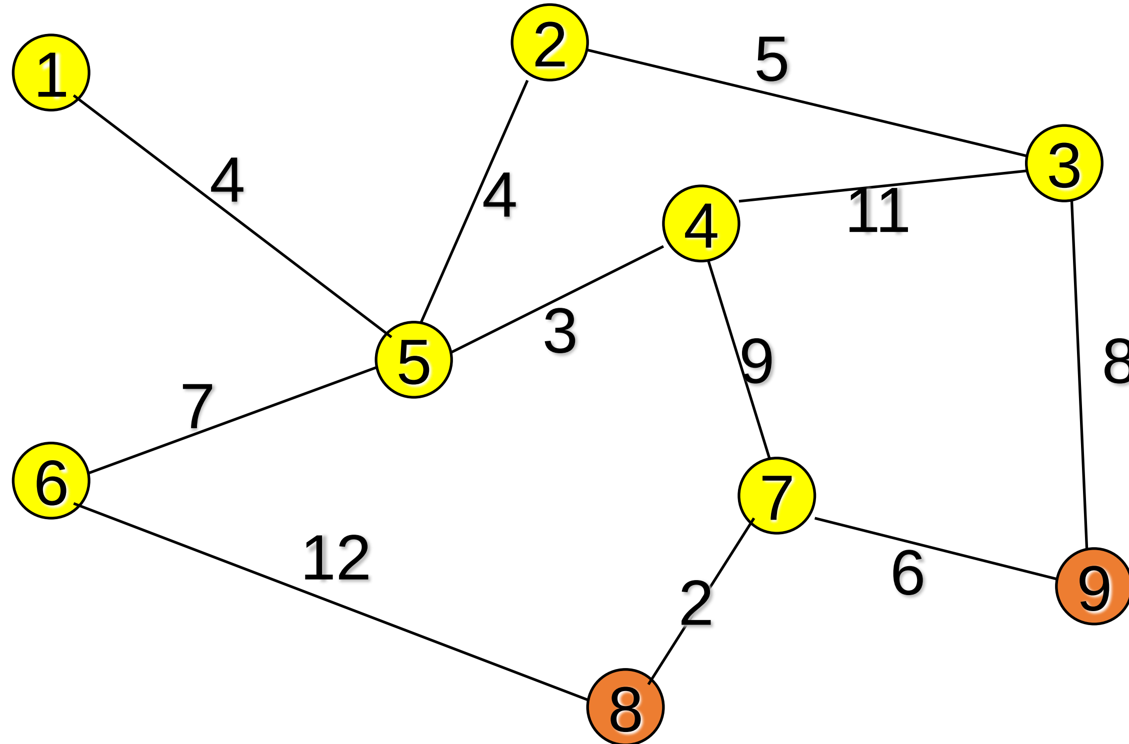
Example of Dijkstra's algorithm

- مثال (2):



Processed	1	2	3	4	5	6	7	8	9
distance	0	8	13	7	4	11	16	23	21
predecessor	-1	5	2	5	1	5	4	6	3

Example of Dijkstra's algorithm

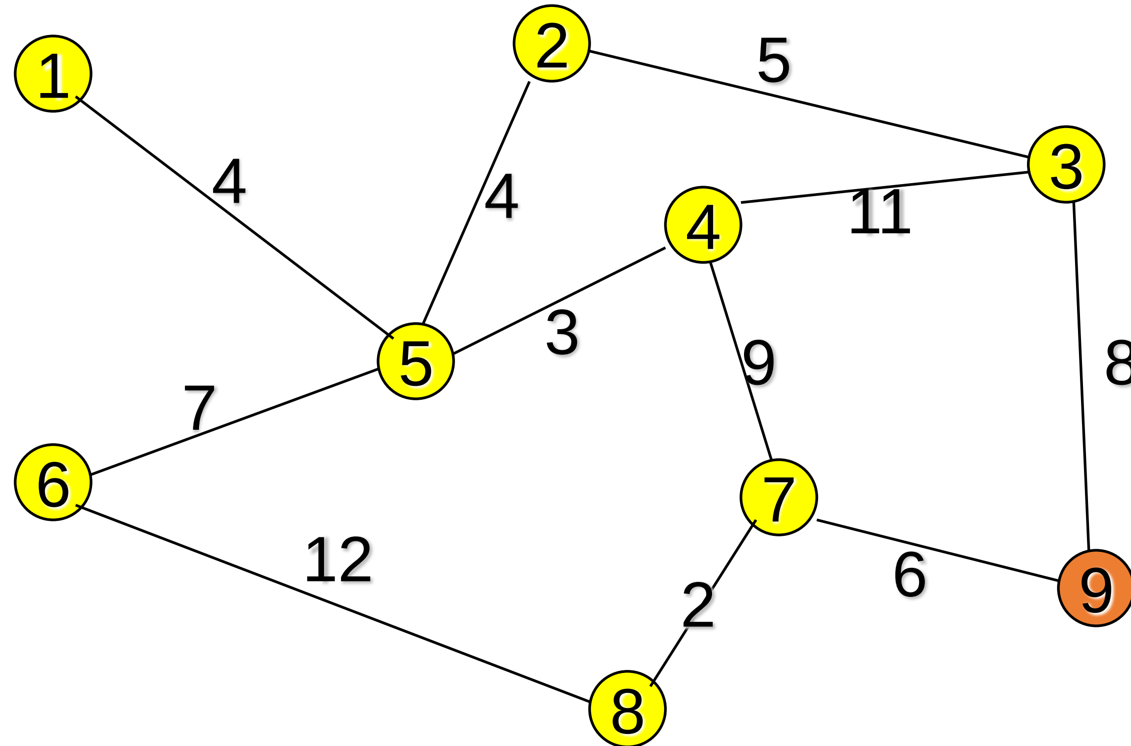


- مثال (2):

Processed	1	2	3	4	5	6	7	8	9
distance	0	8	13	7	4	11	16	18	21
predecessor	-1	5	2	5	1	5	4	7	3

Example of Dijkstra's algorithm

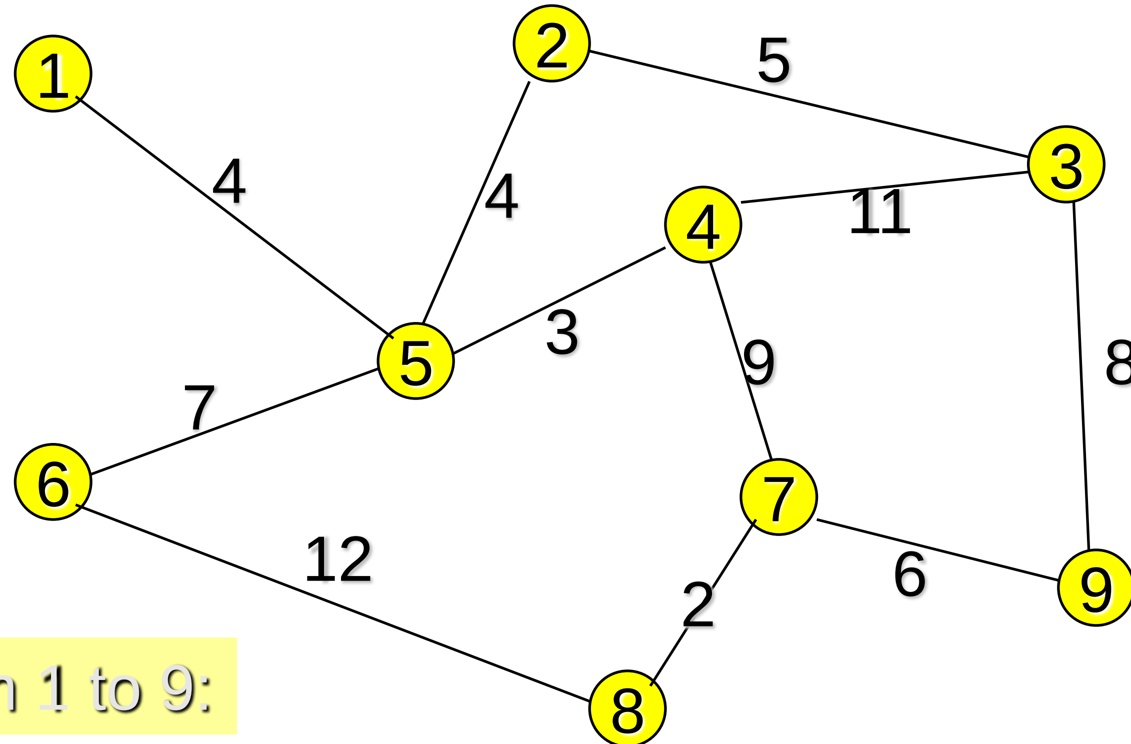
- مثال (2):



Processed	1	2	3	4	5	6	7	8	9
distance	0	8	13	7	4	11	16	18	21
predecessor	-1	5	2	5	1	5	4	7	3

Example of Dijkstra's algorithm

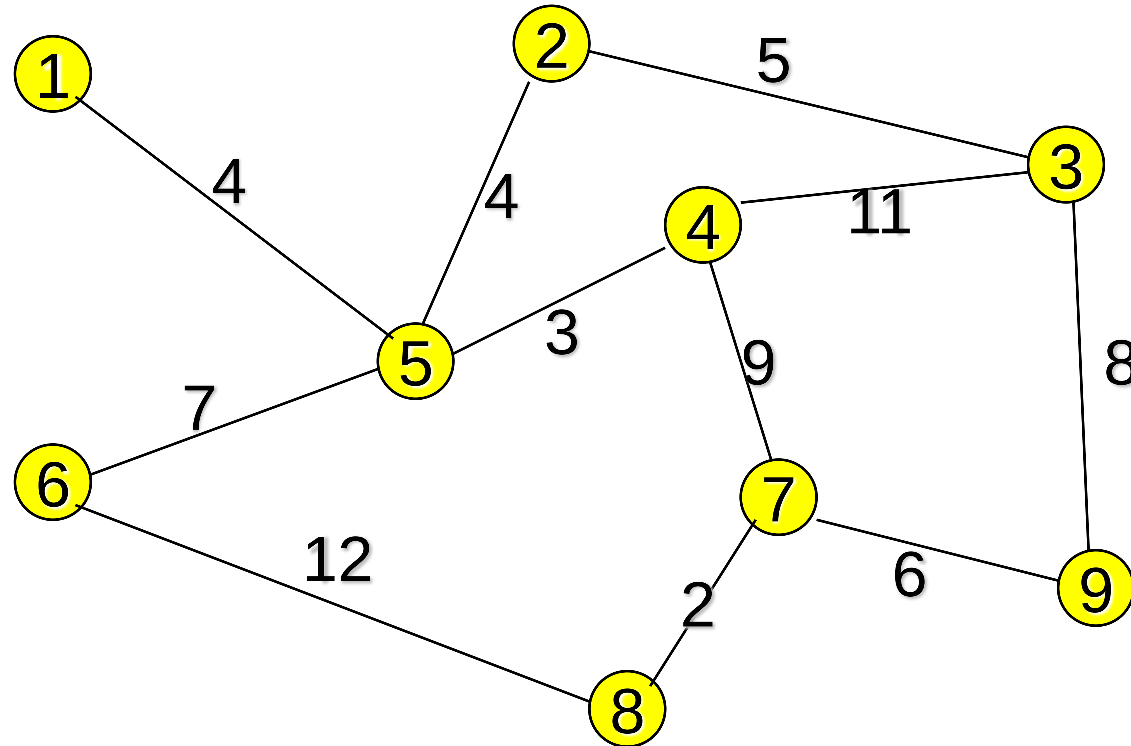
- مثال (2):



Path from 1 to 9:

Processed	1	2	3	4	5	6	7	8	9
distance	0	8	13	7	4	11	16	18	21
predecessor	-1	5	2	5	1	5	4	7	3

Example of Dijkstra's algorithm

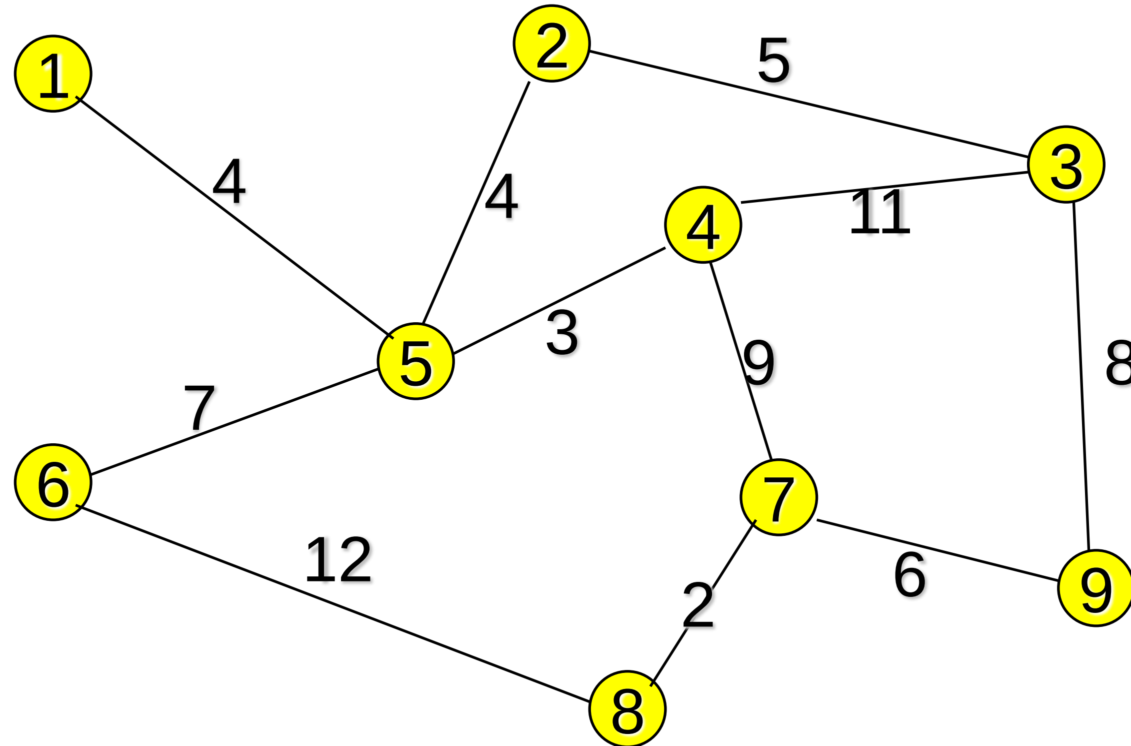


- مثال (2):

Processed	1	2	3	4	5	6	7	8	9
distance	0	8	13	7	4	11	16	18	21
predecessor	-1	5	2	5	1	5	4	7	3

Example of Dijkstra’s algorithm

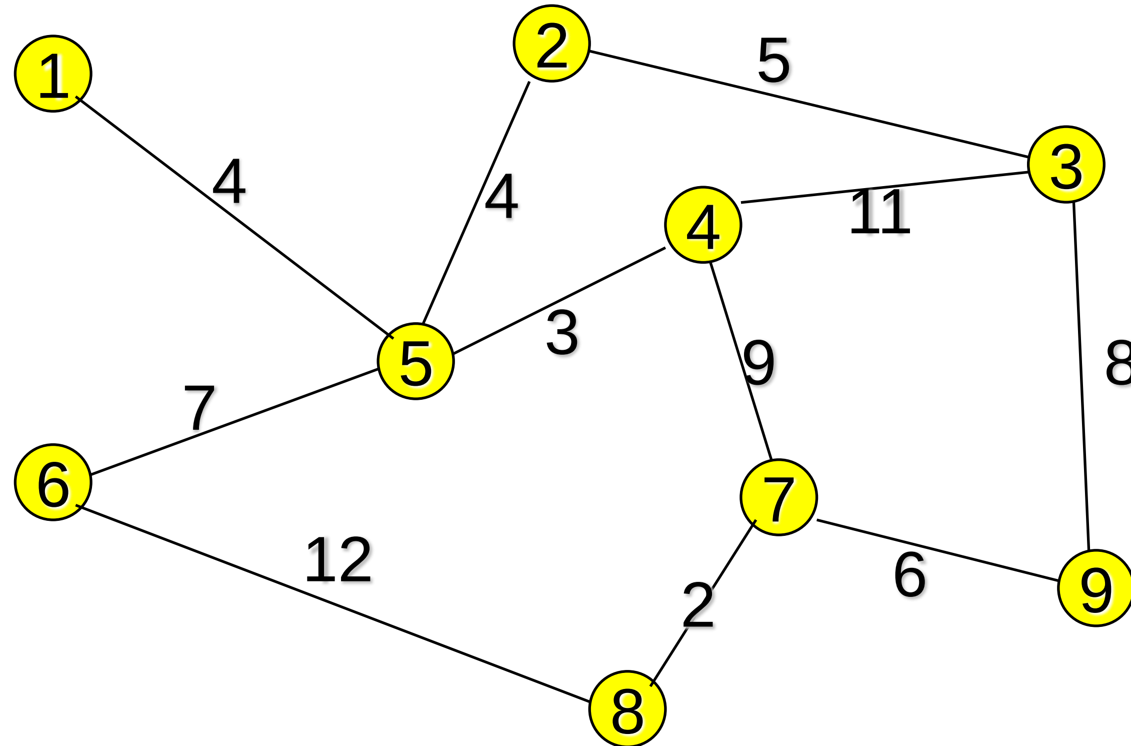
▪ مثال (2):



Processed	1	2	3	4	5	6	7	8	9
distance	0	8	13	7	4	11	16	18	21
predecessor	-1	5	2	5	1	5	4	7	3

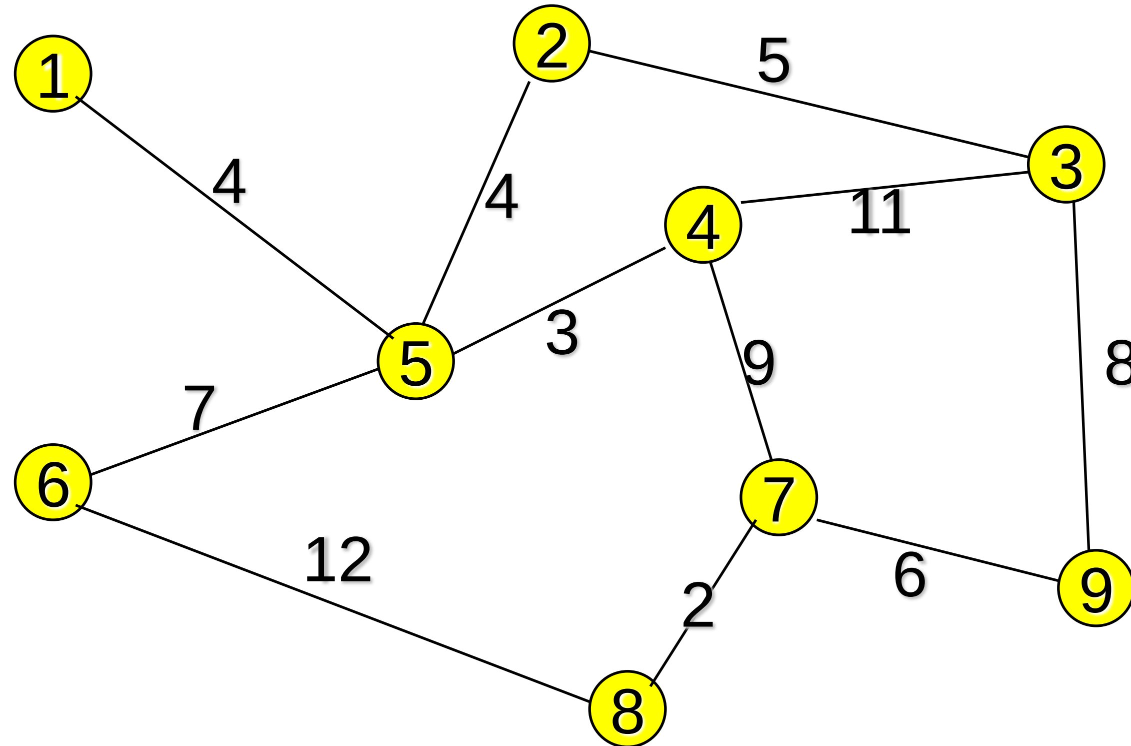
Example of Dijkstra's algorithm

- مثال (2):



	1	2	3	4	5	6	7	8	9
Processed									
distance	0	8	13	7	4	11	16	18	21
predecessor	-1	5	2	5	1	5	4	7	3

Example of Dijkstra’s algorithm

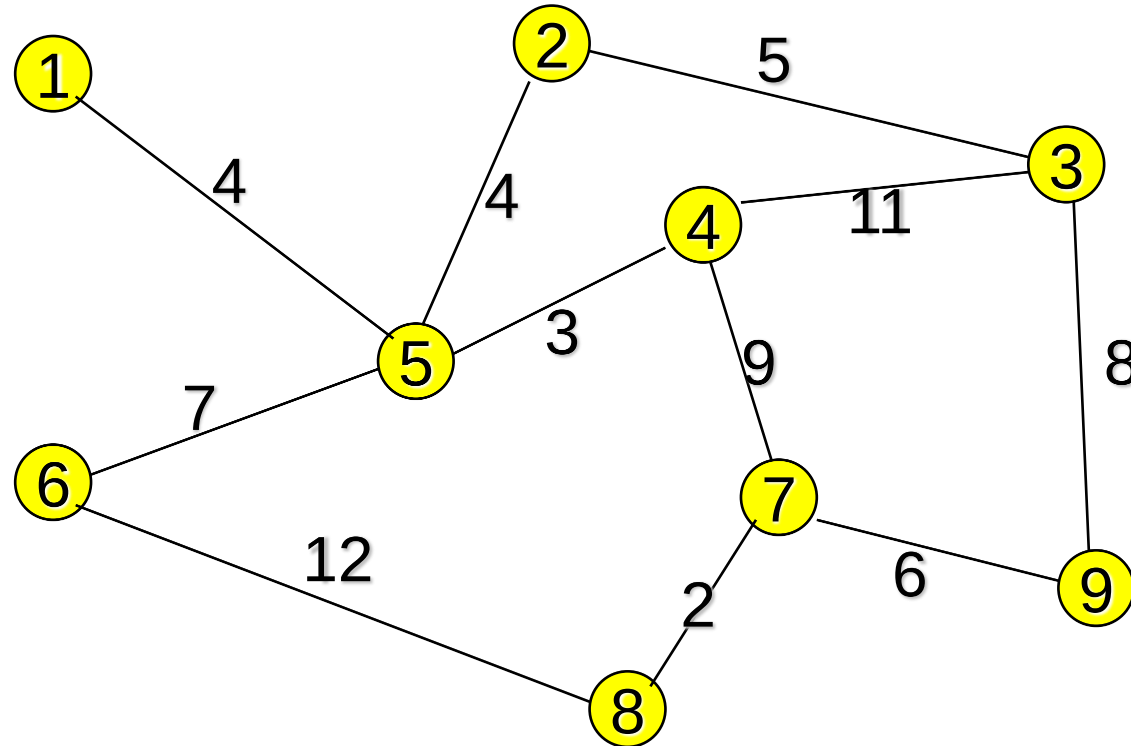


- مثال (2):

	1	2	3	4	5	6	7	8	9
Processed									
distance	0	8	13	7	4	11	16	18	21
predecessor	-1	5	2	5	1	5	4	7	3

Example of Dijkstra's algorithm

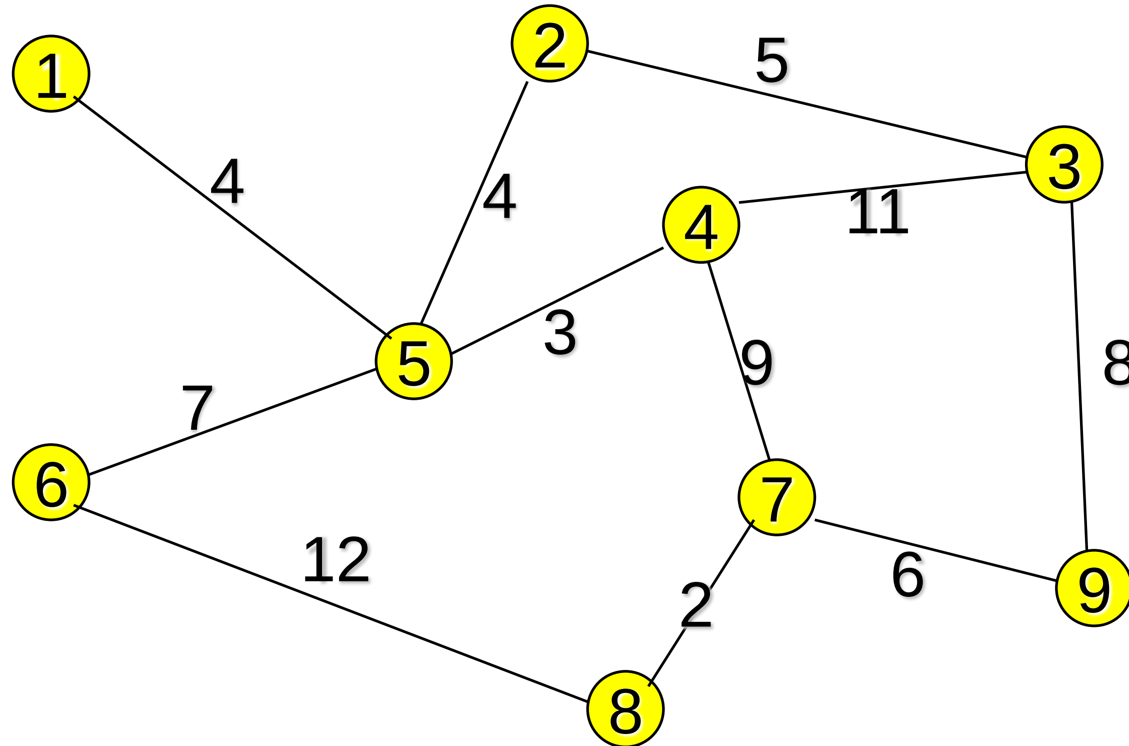
- مثال (2):



Processed	1	2	3	4	5	6	7	8	9
distance	0	8	13	7	4	11	16	18	21
predecessor	-1	5	2	5	1	5	4	7	3

Example of Dijkstra's algorithm

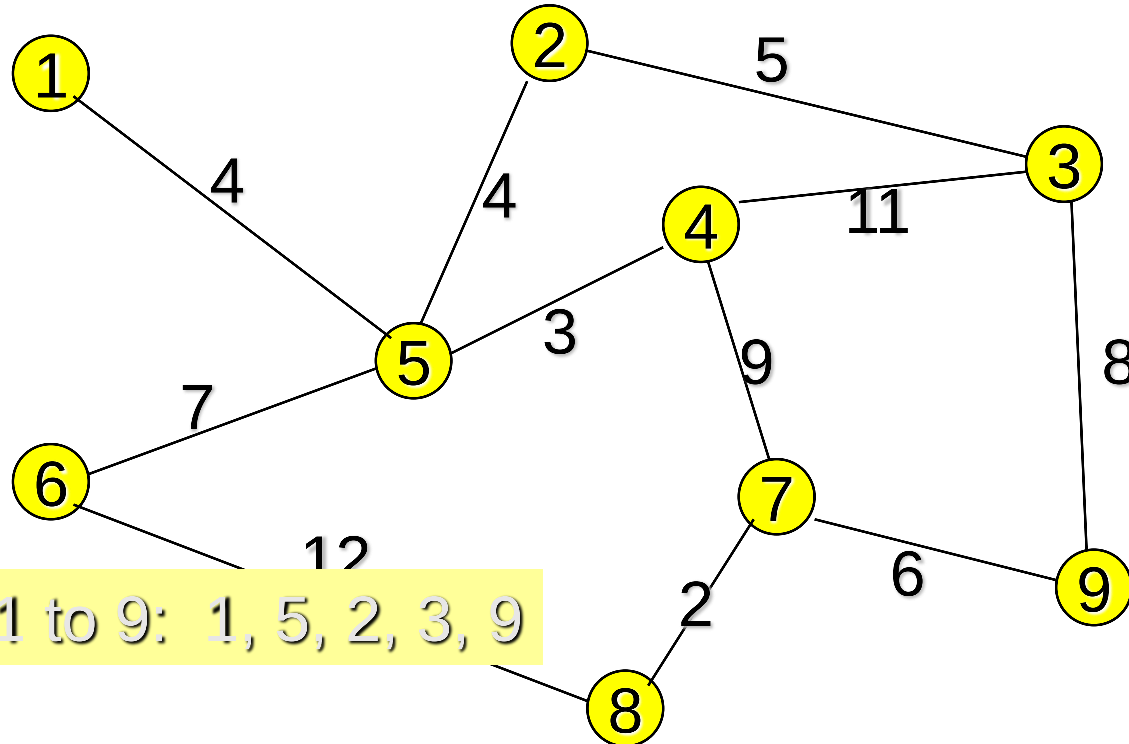
- مثال (2):



	1	2	3	4	5	6	7	8	9
Processed									
distance	0	8	13	7	4	11	16	18	21
predecessor	-1	5	2	5	1	5	4	7	3

Example of Dijkstra’s algorithm

- مثال (2):



	1	2	3	4	5	6	7	8	9
Processed									
distance	0	8	13	7	4	11	16	18	21
predecessor	-1	5	2	5	1	5	4	7	3

تطبيقات خوارزميات (SSSP)

- تستخدم خوارزميات (SSSP) في معظم نظم تحديد المواقع (GPS) ونظم التوجيه (Routing Systems).

